1

MODIFICATIONS TO MCNP, VERSION 3B: INCORPORATING A VARIABLE DENSITY ATMOSPHERE

AFIT TECHNICAL REPORT: AFIT/EN-TR-91-2

Capt David L. Monti, USAF, B.S.
LCDR Kirk A. Mathews, USN, Ph.D.

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/EN-TR-91-2
March 1991

MODIFICATIONS TO MCNP, VERSION 3B: INCORPORATING A VARIABLE DENSITY ATMOSPHERE

AFIT TECHNICAL REPORT: AFIT/EN-TR-91-2

Capt David L. Monti, USAF, B.S.
LCDR Kirk A. Mathews, USN, Ph.D.

PREFACE

This document was written to serve as an addendum to Master's Thesis project AFIT/GNE/ENP/91M-6, "High Altitude Neutral Particle Transport Using the Monte Carlo Simulation Code MCNP With Variable Density Atmosphere". The research and development for this thesis was provided by Captain David L. Monti, USAF. Assistance was provided by Lieutenant Commander Kirk A. Mathews, USN. This document was written to provide understanding and assistance to users who plan to implement additional modifications to MCNP.

This effort was part of an ongoing AFIT research project to improve accuracy, decrease run time, and improve overall Monte Carlo transport methods at AFIT. There were two primary goals of this thesis project. The first was to develop an accurate model describing the variation of air density with altitude and to incorporate it into a generalized version of the Monte Carlo code MCNP. The second goal was to perform radiation transport simulations using a point isotropic neutron-photon source and study the effects over long ranges.

This work was sponsored by the Air Force Weapons Laboratory; this AFIT technical report is intended to serve as a formal addendum to the actual thesis project. A primary goal in this report is to provide the sponsor and all future users of the modified MCNP code with detailed information regarding the additions and modifications made to MCNP, version 3B. We hope it provides the necessary documentation to assist in future work in this area.

Table of Contents

ABSTRACT

MCNP version 3B was modified to incorporate a continuously variable density atmosphere. This was accomplished by representing the variation of air density as a function of altitude with a series of continuous piecewise exponential curves up to a maximum altitude of 1000 km. User-written subroutines and functions were written which incorporated these piecewise functions. These subroutines and functions were subsequently incorporated into a production version of MCNP. Several MCNP subroutines and files were modified in support of these modifications. This report discusses detailed information regarding the theoretical development of the variable density model, the user-written subroutines and functions, the modifications to MCNP subroutines and files, and other relevant information.

1 Introduction

MCNP version 3B [1] was modified to incorporate a continuously variable density atmosphere as part of a Master's Thesis project. This addendum is part of Master's Thesis AFIT/GNE/ENP/91M-6 [2], still unpublished at this writing. There were several reasons for incorporating these modifications. These include: 1) improving the accuracy of computed Monte Carlo density-dependent radiation transport results within the atmosphere, 2) decreasing the overall computer run time, 3) increasing the dimensions of the problem geometry, and 4) lessening the user overhead in preparing the input files. A full discussion on this topic can be found in Master's Thesis AFIT/GNE/ENP/91M-6 by David L. Monti.

There are several areas requiring further work on the modified MCNP code, some of which were discussed in Section VIII of AFIT/GNE/ENP/91M-6. This addendum was written to assist future users in performing additions and/or modifications to the modified MCNP version 3B code at the Air Force Institute of Technology (AFIT).

1.1 Background

The purpose of the thesis project was to perform Monte Carlo simulations of neutral particle transport with primary and secondary photon production as it applies to a point isotropic source within a variable density atmosphere. Specifically, the goal was to increase the accuracy of neutral particle transport calculations within the atmosphere by accurately representing the variability of air density with altitude. In previous studies of Monte Carlo radiation transport within the atmosphere, many discrete spatial cells were used to represent the change in air density as a function of altitude. This was accomplished by using a series of constant density regions, usually with a constant scale height factor. This proliferation of cells proved to be costly in terms of computer run time, especially for large dimension problems.

During this study, analytic representations for air density and mass integral which modelled the variation of air density with altitude were developed. The generalized Monte Carlo code MCNP, version 3B, was modified using the newly developed atmospheric model. The work included the development of user-written subroutines and functions as well as modification of MCNP subroutines and files.

1.2 Approach to Problem

The approach to the problem of developing a variable density atmosphere was based on a two-phase approach: 1) develop analytic expressions to represent both the variation of air density with altitude and cumulative mass-integral with altitude, and 2) implement these analytic functions into MCNP via user-written subroutines and functions. It was also necessary to perform auxiliary modifications to MCNP files and subroutines in support of the variable density model.

2 Project Goals

The overall objectives of the thesis project were as follows:

1. Develop a working Monte Carlo simulation code (MCNP) that incorporates a continuously variable air density capability;

2. Improve the accuracy of density-dependent transport results over previous results;

3. Decrease computer run time relative to the unmodified Monte Carlo code, MCNP;

4. Increase the capability to run very large dimension problems;

5. Study the radiation effects from a high altitude burst over long ranges.

The overall objectives of this report are as follows:

1. Provide detailed information on the atmospheric density model;

2. Provide detailed information on the user-written subroutines and functions incorporated into MCNP;

3. Provide detailed information on MCNP files and subroutines modified in support of the variable density atmosphere.

3 Sequence of Presentation

Section 4 presents the relevant MCNP program flow. Section 5 provides the theoretical discussion behind the variable density representation of the atmosphere. Section 6 presents a detailed discussion of the user-written subroutines and functions integrated into MCNP as part of the modifications. Section 7 provides a detailed discussion of the affected MCNP subroutines. Section 8 presents some miscellaneous information concerning the MCNP code. Appendix A contains listings of the user-written subroutines and functions integrated into MCNP. Appendix B contains listings of the MCNP files and subroutines that were modified.

4 MCNP Program Flow

This section describes the relevant portions of MCNP program flow. Topics discussed include the program flow of Subroutines HSTORY and TRANSM, where most of the particle transport takes place.

4.1 Transport Program Flow

Particle transport in MCNP is controlled via the main transport overlay MCRUN. MCRUN calls subroutine TRNSPT to start particle histories via subroutine HSTORY.

4.1.1 Subroutine HSTORY

Subroutine HSTORY begins the particle history by starting a particle from the source via subroutine STARTP. All source parameters such as initial direction, weight, and energy are determined. Initial source parameters are next checked against user-supplied weight cut-offs and energy cut-offs and some summary information is incremented. Subroutine TALLY is called to score tally contributions during a particle history. If point detectors or ring detectors are being used, then subroutine TALLYD is called from TALLY to score contributions to detectors. If DXTRAN spheres are being used, then DXTRAN is called to score DXTRAN sphere contributions. The weight window variance reduction game is played if requested, and energy splitting or Russian roulette is performed if required.

Back in subroutine HSTORY, particle transport continues by calling TRACK to determine the intersection of the particle trajectory with each bounding surface of the cell. The distances to the nearest cell boundary, DLS, time cut-off, DTC, and DXTRAN sphere, DXL, are calculated. Cross sections for the cell are determined via subroutines ACETOT for neutrons and PHOTOT for photons. A macroscopic cross section, QPL, is calculated and then modified by the exponential transform in subroutine EXTRAN, if necessary. The distance to next collision, PMF, is determined by a random number sampling of a logarithmic distribution. Up to this point, no modifications were made to subroutine HSTORY.

The track length in the cell is determined by taking the minimum of the distance to collision, PMF, the distance to time cut-off, DTC, the distance to the nearest bounding surface, DLS, and the distance to a DXTRAN sphere, DXL. The particle will undergo a collision only if the distance to collision, PMF, is less than the distance to a surface crossing, DLS. Subroutine TALLY is called to increment any cell tallies if necessary and some summary information is incremented. The particle's position is also updated. The energies and directions of the particles exiting a collision site are handled in subroutines ACECAS and ACEOS.

4.1.2 Subroutine TRANSM

Subroutine TRANSM is called from TALLYD to calculate the total transmission to the detector. Prior to calculating the transmission, subroutine DDDET is called to calculate the distance, DD, to the detector and TRACK is called to calculate the distance to the nearest intersecting surface, DLS.

Cross sections for the cell are determined via subroutines ACETOT for neutrons and PHOTOT for photons. Once the microscopic cross section in the cell has been determined, it is multiplied by the atom density in the cell to obtain the macroscopic cross section. The track length in the cell, D, is determined by taking the minimum of the distance to the nearest bounding surface, DLS, and the distance to the detector, DD. The number of mean free paths to the detector or nearest bounding surface, AMFP, is determined using the values of the macroscopic cross section in the cell, PLE, and the minimum distance to the detector or nearest bounding surface, D. Then the locations of the new cell and particle position are updated.

5 Theoretical Discussion of the Variable Density Atmosphere

This section presents a detailed theoretical discussion of the variable density atmosphere incorporated into MCNP, version 3B. This served as a basis for the development of user-written subroutines and functions which were subsequently incorporated into MCNP. Topics discussed include: definition of a reference density and calculation of important atmospheric transport parameters.

5.1 Reference Density

Modelling the variable density atmosphere involved defining a reference air density. In the unmodified version, MCNP computes density-dependent parameters using a discrete cell density defined in the input file. In the modified version, the variation of air density is continuous, and a constant density spatial mesh in the vertical direction is no longer needed, except for splitting/Russian roulette. Therefore, in the modified version of MCNP, the air density in each cell is redefined to be the mass density at sea-level, 1.225×10^{-3} g/cm^3, regardless of the values read from the input file. All subsequent calculations of density-dependent parameters will now be calculated based on a sea-level homogeneous atmosphere. At this point, prior to computing the distance to collision or the transmission to the detector or nearest intersecting boundary, the MCNP calculations are intercepted.

The aforementioned parameters are then corrected for a variable density atmosphere.

Prior to running the MCRUN overlay, the main overlay IMCN is run to read the input file to obtain the necessary parameters used in defining the problem. Subroutine NEXTIT is called to process items from the input file, i.e., to store such parameters as material density and cell descriptions. Subroutine NEXTIT was modified for the purpose of redefining the density in each cell to be the density at sea-level.

5.2 Transport Parameters

In Monte Carlo transport calculations, there are two principle parameters that involve determining the optical path length between two points in space: 1) the sampled distance to collision and 2) he number of mean free paths to a detector or nearest intersecting boundary. Both parameters are closely related and depend strongly on the atmospheric density. Details of the calculation of each of these two important transport parameters is presented below.

5.2.1 Distance To Collision

The distance to collision, PMF, computed in HSTORY, is based on the microscopic cross section in the cell and the atom density in the cell, which was redefined in NEXTIT to be the air density at sea-level. This is given by

$$PLE = TOTM * DEN(CELL) \tag{1}$$

where

PLE = macroscopic cross section in current cell [cm^{-1}]
TOTM = microscopic cross section in current cell for either a neutron or photon reaction [barns/atom]
DEN(CELL) = atom density in current cell [atoms/barn-cm]

Since the distance to collision is a probability based on density, it is necessary to correct this value for a variable density atmosphere. Subroutine EQDIST, called from HSTORY, was written to calculate the corrected distance to collision based on values input from HSTORY, i.e., current particle altitude, z-direction cosine, and distance to collision based on a sea-level homogeneous atmosphere. Calculation of the required quantities used to determine the corrected distance to collision are explained below. See Figure 3, Section III, AFIT/GNE/ENP/91M-6, "Mass-Integral Scaling (MIS)" for the general coordinate geometry used in this analysis. The variable names referenced in the discussion below are the variable names used in the non-MCNP (user-supplied) subroutines or functions.

The unmodified version of MCNP computes distances in units of distance, centimeters. However, the variable density atmospheric model computes distances in units of mass range, g/cm². Therefore, before correction for a variable density atmosphere can be made, it was first necessary to transform the MCNP-computed distance to collision, PMF (EDST in subroutine EQDIST), from units of centimeters to an equivalent mass range in units of g/cm² for a sea-level homogeneous atmosphere. This equivalent distance was then corrected for a variable density atmosphere using the MCNP modifications.

The mass-integral along the particle path in a homogeneous sea-level atmosphere is calculated using the following equation

$$MIPDH = 1.225x10^{(-3)} EDIST \qquad (2)$$

where

MIPDH = equivalent mass range along particle path in a sea-level homogeneous atmosphere [g/cm²]
EDIST = distance to collision in a sea-level homogeneous atmosphere [cm]

This calculation is performed in function DMI.

The cumulative mass-integral, MICA, at the current particle altitude, ZO, can be computed using Equation (3) below once the scale height region has been determined. This calculation is performed in function MASI.

$$MICA = MI(z_i) + \rho(z_i) S_i \left[1 - \exp\left(-\frac{(z0 - z_i)}{S_i}\right)\right] \qquad (3)$$

where

MICA = mass-integral at current particle altitude [g/cm²]
MI(z_i) = mass-integral at base altitude of ith region [g/cm²]
$\rho(z_i)$ = density at base altitude of ith region [g/cm³]
z0 = current particle altitude [cm]
z_i = base altitude of ith region [cm]

S_i = mass-integral scale height factor of ith region [cm]

$z_i \le z0 \le z_{i+1}$

Though the collision altitude, ZC, is still unknown, the values of MIPDH, MICA, and the z-direction cosine, WO, can be used to determine the required cumulative mass-integral, MICP, at the collision altitude, ZC. MICP is computed based on the known mass range to collision, MIPDH. This calculation is performed by calling function MIZ.

$$MICP = MICA + MIPDH * WO \tag{4}$$

where

MICP = required mass-integral at collision altitude [g/cm²]

MICA = mass-integral at current particle altitude [g/cm²]

MIPDH = mass-integral along particle path to collision in a sea-level homogeneous atmosphere [g/cm²]

WO = z-direction cosine

At this point, a check is made to compare the computed cumulative mass-integral at the collision altitude, MICP, with the mass-integral at infinity, MIINF, defined to be 1035.635131402448 g/cm², the cumulative mass-integral at the 990 km altitude limit. If MICP is greater than MIINF, then Subroutine EQDIST is exited and PMF is set to HUGE, a very large number used by MCNP, effectively eliminating any collisions along the current particle track. If MICP is less than or equal to MIINF, then the collision altitude, ZC, is determined using the following expression

$$zc = z_i - S_i \ln\left[1 + \frac{(MI(z_i) - MI(zc))}{(\rho(z_i) S_i)}\right] \tag{5}$$

For cases where the z-direction cosine, WO, becomes very small (nearly co-altitude relative to the current particle position), the change in density is very small along the particle path. Therefore, the average density between the current altitude and the collision altitude is used. In this case, the difference between the current altitude and collision altitude is ≤ 10 meters. The air densities are calculated by calling function DNTY.

Finally, the distance to collision, PMF (EDST in Subroutine

EQDIST), corrected for a variable density atmosphere, can be computed by calling function EQUIVDST. Function EQUIVDST accepts as inputs the collision altitude, ZC, the current particle altitude, ZO, the z-direction cosine, WO, the mass-integral along the particle direction in a homogeneous sea-level atmosphere, MIPDH, and the densities (calculated only if WO is small). For cases where WO is not small, the corrected distance to collision is given by Equation (6) below

$$EDST=\frac{(ZC-ZO)}{WO} \tag{6}$$

where

EDST = distance to collision corrected for a variable density atmosphere [cm]

For cases where WO is small, the density is nearly constant along the particle path, hence the corrected distance to collision is given by

$$EDST=\frac{MIPDH}{[(DENCA+DENCP)/2]} \tag{7}$$

where

DENCA = air density at current particle altitude [g/cm^3]
DENCP = air density at collision altitude [g/cm^3]

5.2.2 Mean Free Paths To Detector or Boundary

Contributions to a detector are made at every source or collision point by creating and transporting a pseudoparticle directly to the detector. The total transmission to the detector depends on several factors: 1) the exponential attenuation through the medium, 2) a probability density function for scatter toward the detector, 3) spherical divergence, which accounts for the solid angle effect, and 4) the particle weight.

The only one of the aforementioned parameters that depends on atmospheric density is the exponential attenuation term. The attenuation term represents the total attenuation of the radiation by the atmosphere along the particle path over a given number of mean free paths. The attenuation along the particle path to a detector is given by the following relation

$$A = \exp(-AMFP) \quad (8)$$

where

A = attenuation along the particle path
AMFP = number of mean free paths to the detector or nearest intersecting boundary

At this point, MCNP computes the distance to the detector, DD, using subroutine DDDET. MCNP next determines the minimum of the distances to the detector or the nearest intersecting boundary. MCNP now computes the macroscopic cross section in the cell, PLE, for a sea-level homogeneous atmosphere using Equation (1). It is necessary to correct the macroscopic cross section in the cell, PLE, for a variable density atmosphere using subroutine EQDIST. A flag is set prior to calling EQDIST in subroutine TRANSM to indicate the origin of the calling statement, either subroutine HSTORY or subroutine TRANSM. The value of the flag, either 0 or 1, will determine the specific calculations that are performed.

Prior to calling EQDIST, the reciprocal of the macroscopic cross section, XMFP = 1/PLE, is computed to give the mean free path in a sea-level homogeneous atmosphere (recall that MCNP first calculates all density-dependent parameters based on sea-level density in each cell). EQDIST is then called from subroutine TRANSM with the mean free path, XMFP, the current altitude, ZZZ, the z-direction cosine, WWW, the calculation flag, NINP, and the track length in the cell, D, as inputs.

In subroutine EQDIST, it is first necessary to transform the MCNP-computed minimum distance to the detector or nearest intersecting boundary, D, from units of centimeters to an equivalent mass range in units of g/cm^2 in a sea-level homogeneous atmosphere. This equivalent mass range is then used to correct the macroscopic cross section, PLE, for a variable density atmosphere.

In subroutine EQDIST, the altitude of the detector or nearest intersecting bounding surface is determined using the following equation

$$ZD = ZO + WO * DTD \quad (9)$$

where

ZD = altitude of detector or intersecting boundary [cm]
ZO = current particle altitude [cm]
WO = z-direction cosine
DTD = distance to detector or intersecting boundary [cm]

The mass-integral along the particle path in a homogeneous atmosphere, MIPDH, is next calculated using Equation (2). Function DMI performs this calculation.

Next, the cumulative mass-integral at the current particle altitude, MICA, and the cumulative mass-integral at the detector or intersecting boundary altitude, MIDA, is computed using Equation (3). This calculation is performed in function MASI.

The mass-integral along the particle path in a variable density atmosphere, MIPD, can now be determined using the following relation (for WO not too small)

$$MIPD = \frac{(MIDA - MICA)}{WO} \tag{10}$$

where

MIPD = mass-integral along the particle path in a variable density atmosphere [g/cm²]

For cases where the z-direction cosine, WO, becomes very small (nearly co-altitude relative to the current particle position), the average density between the current altitude and the collision altitude is used. In this case, the difference between the current altitude and collision altitude is ≤ 10 meters. The air densities are calculated by calling function DNTY. For small values of the z-direction cosine, WO, MIPD is computed using the following relation

$$MIPD = DTD * \left[\frac{(DENCA + DENCP)}{2} \right] \tag{11}$$

where

DTD = minimum distance to the detector or nearest intersecting boundary [cm]
DENCA = air density at current particle altitude [g/cm³]
DENCP = air density at collision altitude [g/cm³]

Recall that the minimum of the distances to the detector and nearest intersecting boundary, D, was first transformed into an equivalent mass range, MIPDH. The corrected mass range in a variable density atmosphere, MIPD, was then computed. It is now possible to correct the mean free path to the detector or nearest intersecting boundary, EDST, for a variable density atmosphere. This can be calculated using

$$EDST=\frac{(EDST*MIPDH)}{MIPD} \tag{12}$$

where

EDST [LHS] = corrected mean free path [cm]
EDST [RHS] = uncorrected mean free path [cm]
MIPDH = mass range along particle path in a sea-level homogeneous atmosphere [g/cm^2]
MIPD = mass integral along particle path in a variable density atmosphere [g/cm^2]

Back in TRANSM, the corrected macroscopic cross section in the cell, PLE, is found by taking the reciprocal of the corrected mean free path, EDST (defined as XMFP in MCNP), where EDST is given by Equation (12). Finally, the cumulative total number of mean free paths to the detector or nearest boundary over all cells traversed by the particle trajectory, is found from the following equation

$$AMFP=AMFP+PLE*D \tag{13}$$

where

AMFP [LHS] = cumulative total number of mean free paths to the detector or nearest boundary along the particle path
AMFP [RHS] = number of mean free paths along the particle path in the current cell

AMFP here is equal to PLE*D.

6 User-Written Subroutines and Functions

This section describes in detail the implementation of the variable density model developed in Section 5. This was accomplished by developing various user-written subroutines and functions and later integrating them into MCNP. The variables referenced in the discussion below refer to the variable names used in the user-written subroutines or functions as opposed to the variable names used in the MCNP subroutines.

6.1 Data Blocks

There was only one data block defined as part of the MCNP modifications and is described below.

6.1.1 Data Block ATINIT

ATINIT is a user-defined data block which initializes the arrays containing parameters which define the variable density atmospheric model. The arrays are stored in a user-defined common block /ATCOM/. Common block /ATCOM/ is defined in the file CM.inc.

There are 5 arrays contained in data block ATINIT. The arrays SHFM and SHFD contain scale height factors [cm] for both mass-integral [g/cm^2] and density [g/cm^3] data, respectively. The array ZI contains the base altitude [cm] for each scale height region. The array DENI contains the reference density [g/cm^3] at each base altitude. The array AMII contains the cumulative mass-integral [g/cm^2] at each base altitude.

6.2 Functions

There were 6 user-written functions incorporated into the MCNP code. The function of each of these is explained below.

6.2.1 Function DNTY

This function calculates a density, DNTY [g/cm^3], given an input altitude, Z [cm]. Function DNTY is called from subroutine EQDIST. In order to compute a density based on the variable density atmospheric model, the array ZI containing base altitudes for each scale height region, the array DENI containing densities at each base altitude, and the array SHFD containing density scale height factors for the scale height regions are required. Function DNTY first determines the correct scale height region based on the input altitude, Z, by searching the base altitude array, ZI. The array indexing variable, k, is used to index the DENI and SHFD arrays to obtain the corresponding density and scale height factor for that scale height region. Function DNTY then computes the density at Z using the correct atmospheric model parameters.

6.2.2 Function MASI

This function calculates a mass-integral, MASI [g/cm^2], given an input altitude, Z [cm]. The computed mass-integral represents the cumulative mass-integral from ground level up to the input altitude, Z. Function MASI is called from subroutine EQDIST. In order to compute the cumulative mass-integral based on the variable density atmospheric model, the array ZI containing base altitudes for each scale height region, the array DENI containing densities at each base altitude, the array SHFM containing density scale height factors for the scale height regions, and the array AMII containing the cumulative mass-integrals for the base altitudes are required. Function MASI first determines the correct scale height region based on the input altitude, Z, by searching the base altitude array, ZI. The array indexing variable, k, is used to index the DENI, SHFM, and AMII arrays to obtain the corresponding density, scale height factor, and cumulative mass-integral for that scale height region. Function MASI then computes the density at Z using the correct atmospheric model parameters.

6.2.3 Function ZMAS

This function calculates an altitude, ZMAS [cm], given an input mass-integral, MI [g/cm^2]. The computed altitude represents the altitude necessary to achieve the given cumulative mass-integral. Function ZMAS is called from subroutine EQDIST. In order to compute the required altitude based on the variable density atmospheric model, the array ZI containing base altitudes for each scale height region, the array DENI containing densities at each base altitude, the array SHFM containing mass-integral scale height factors for the scale height regions, and the array AMII containing the cumulative mass-integrals for the base altitudes are required. Function ZMAS first determines the correct scale height region based on the input mass-integral, MI, by searching the array, AMII. The AMII array contains the cumulative mass-integral for the base altitudes of the scale height regions. The array indexing variable, k, is used to index the DENI, SHFM, and ZI arrays to obtain the corresponding density, scale height factor, and base altitude for that scale height region. Function ZMAS then computes the altitude using the correct atmospheric model parameters.

6.2.4 Function DMI

This function calculates a mass-integral, DMI [g/cm^2], between two points in a sea-level homogeneous atmosphere. The required input value is: 1) the distance to collision or mean free path to the boundary or detector [cm]. Function DMI is called from subroutine EQDIST. Function DMI computes the mass range along the particle path of known range in a sea-level homogeneous atmosphere. The mass density at sea-level is given by the U.S. Standard Atmosphere, 1976 [3], to be 1.225×10^{-3} g/cm^3.

6.2.5 Function MIZ

This function calculates a mass-integral, MIZ [g/cm^2], at the new particle altitude in a variable density atmosphere. The new particle altitude is not yet explicitly known. The required input values are: 1) the mass-integral at the current particle altitude, MIP [g/cm^2], 2) the mass range along the particle path in a sea-level homogeneous atmosphere, DMI [g/cm^2], and 3) the z-direction cosine, W. The z-direction cosine represents the cosine of the angle between the polar axis, Z, and the horizontal axis, Y. Function MIZ is called from subroutine EQDIST. Function MIZ computes the mass-integral required at the new particle altitude based on previously computed mass-integral values and the current z-direction cosine. The new particle altitude is computed subsequently.

6.2.6 Function EQUIVDST

This function calculates an equivalent distance to collision, EQUIVDST [cm], along the particle path which gives the same mass-integral as computed in a sea-level homogeneous atmosphere. The required input values are: 1) the current particle altitude, Z [cm], 2) the new particle altitude, ZN [cm], 3) the z-direction cosine, W, 4) the density at the current particle altitude, DCA [g/cm^3], 5) the density at the new particle altitude, DCP [g/cm^3], and 6) the mass range along the particle path in a sea-level homogeneous atmosphere, MIPD [g/cm^2]. Function EQUIVDST is called from subroutine EQDIST. If the difference between the new altitude and the current altitude is less than 1000 cm (10 meters), then W is very small and the density changes very little along the particle path. In this case, the average of the densities at the new altitude and the current altitude are used in computing the new equivalent distance to collision, EQUIVDST. If the difference between the new altitude and the current altitude is greater or equal to 1000 cm (10 meters), then the difference in the altitudes divided by the direction cosine is used to compute EQUIVDST.

6.3 Subroutines

There was only one subroutine written to support the modifications built into MCNP. This subroutine used all the functions described above. This subroutine is described below.

6.3.1 Subroutine EQDIST

This subroutine is used to calculate one of two transport parameters, depending on the calling subroutine within MCNP. If the input value is the sampled distance to collision, PMF

[cm], from subroutine HSTORY, then subroutine EQDIST returns an equivalent distance to collision corrected for a variable density atmosphere, EDST [cm]. If the input value is the calculated mean free path, XMFP [cm], to the detector or nearest intersecting boundary from subroutine TRANSM, then subroutine EQDIST returns an equivalent mear free path corrected for a variable density atmosphere. If subroutine EQDIST is called from subroutine HSTORY, then the input values are: 1) the distance to collision, EDST [cm], 2) the current particle altitude, ZO [cm], 3) the z-direction cosine, WO, 4) the flag NINP. The flag NINP has either the value 0 or 1, indicating which calculations are to be performed. NINP has the value 0 if the calling subroutine is HSTORY, and hence the corrected distance to collision is required. NINP has the value 1 if the calling subroutine is TRANSM, and hence the corrected mean free path to the detector or nearest intersecting boundary is required.

Before any specific calculations are performed, subroutine EQDIST first defines the mass-integral at infinity, MIINF, to be 1035.635131402448 g/cm^2. This represents the cumulative mass-integral at the upper altitude of the atmospheric model, 990 km. Transport calculations are not performed above this altitude limit.

Next, the optimum array search parameters are determined based on the current particle location and direction. Either the upper half, the lower half or the entire array is searched. This decreases the table look-up time versus searching the entire array every time. The array search parameters are passed as input values to the six user-written functions described previously.

At this point, a test is performed to check the value of the flag NINP. Recall that a value of 0 indicates the calling subroutine is HSTORY, in which case the first set of calculations are performed. These calculations are primarily a set of calling statements to one or more of the user-written functions. Each subsequent line of code is well documented as to its function, and thus will not be repeated here. The theory behind each calculation is described in Section 5 of this addendum, and details of the functions performing these calculations are described above under the heading "Functions". It is important to note that before the new collision altitude, ZC [cm], is computed, a test is performed to ensure that the mass-integral required at the collision altitude, MICP, is within both the upper (MIINF) and lower (0.0 g/cm^2) limits of the atmospheric model.

If this test is successful, ther calculations continue. The new corrected distance to collision, EDST, is computed and returned to subroutine HSTORY. In subroutine HSTORY, PMF is defined to be EDST. If the test fails, then the new corrected distance to collision, EDST [cm], is returned with the value -1. In subroutine HSTORY, EDST, with a value of -1, is defined as the variable PMF, and then PMF is set equal to HUGE, a very large number in MCNP. The particle is thus assumed to exit the problem, effectively eliminating any further particle interactions along the path.

If the flag has the value 1, then subroutine EQDIST is called from subroutine TRANSM. First, the detector altitude or altitude of the nearest intersecting boundary with the particle path is computed. The given input values of the direction cosine, WO, and the minimum distance to either the detector or the nearest intersecting boundary, DTD, are used. Each subsequent line of code is well documented as to its function, and thus will not be repeated here. The theory behind each calculation is described in Section 5 of this addendum, and details of the functions performing these calculations are described above under the heading "Functions". It is important to note that before any additional calculations are performed, a test is performed to ensure that the detector or boundary crossing altitude, ZD, is within both the upper (990 km) and lower (0.0 km) limits of the atmospheric model. If this test is successful, then calculations continue. If this test fails, then an error message is printed to the output listing. Calculations continue until a new, corrected mean free path, EDST [cm], is computed. EDST is based on the given input mean free path, EDST [cm], the mass range along the path in a sea-level homogeneous atmosphere, MIPDH [g/cm^2], and the mass range in a variable density atmosphere, MIPD [g/cm^2]. EDST is returned to subroutine TRANSM and defined to be XMFP, the variable used in subroutine TRANSM.

7 Modified MCNP Subroutines and Files

This section describes in detail the auxiliary modifications performed on various MCNP subroutines and files. The variables referenced in the discussion below refer to the variable names used in the MCNP subroutines or files as opposed to the variable names used in the user-written subroutines or functions.

7.1 Include Files

There were two include files that were modified, and these are discussed below.

7.1.1 Comdeck CM.inc

This include file serves as the main common block declaration file for all overlays used in MCNP. The following modifications to CM.inc were made:

1. line 2: the five arrays containing parameters for the atmospheric density model are dimensioned as double precision arrays.

2. line 37: common block /ATCOM/ is defined and contains the necessary arrays that store the various parameters defining the variable density atmospheric model.

7.1.2 Comdeck ZC.inc

This include file serves as an auxiliary comdeck file to comdeck CM.inc and defines processor-dependent constants and parameters and also dimensions general constants and I/O unit numbers. The following modifications to ZC.inc were made:

1. line 6: message defined to inform user that the variable density version of MCNP is being used.

2. line 7: error statement called from subroutine EQDIST.

3. line 25: two variables, NR1 and NR2, are defined and used to dimension the parameter arrays for the atmospheric model. NR1 is the number of atmospheric regions and NR2 is the number of scale height factors.

7.2 Subroutines

There were four subroutines that were modified, and these are discussed below.

7.2.1 Subroutine IMCN

This subroutine serves as the main overlay subroutine, and controls the input and sorting of the problem data. The following modifications to the IMCN overlay were made:

1. line 6: character BLNK defined and used in printing messages.

2. lines 112 thru 118: message printed to both the terminal and output listing informing the user that the variable density version of MCNP is being run.

7.2.2 Subroutine NEXTIT

This subroutine controls the processing of input items.

1. lines 34 thru 40: density in all cells redefined to be 1.225×10^{-3} g/cm^3, the mass density of air at sea-level, regardless of input value.

7.2.3 Subroutine HSTORY

This subroutine performs the main transport calculations including the distance to collision, PMF. Starting on line 61, cross sections [barns/atom] for the cell are determined via subroutines ACETOT for neutrons and PHOTOT for photons. On line 75, a macroscopic cross section, QPL [cm^{-1}], is calculated. In the variable density version of MCNP, it is not recommended that the exponential transform be used until a full study is made of the effects to the macroscopic cross section. The mean free path, GS [cm] is next computed on line 81 by taking the reciprocal of the macroscopic cross section, QPL. Up to this point, no modifications had been made to subroutine HSTORY. The following modifications were made to subroutine HSTORY:

1. line 92: random sampling of the logarithmic distribution function slightly modified. The original calculation was as follows

$$qzridg=RANG() \tag{14}$$

$$PMF=-LOG(qzridg)*GS \tag{15}$$

The above calculations were modified as follows

$$qzridg=RANG() \tag{16}$$

$$qzridg1=-LOG(qzridg) \tag{17}$$

$$PMF=qzridg1*GS \tag{18}$$

where

GS = mean free path [cm] based on the macroscopic cross section, QPL
PMF = distance to next collision [cm]

The purpose of this modification was to enable the recalculation of the mean free path, GS, and the macroscopic cross section, QPL. These values then serve as the effective values corrected for a variable density atmosphere. It is possible that these quantities are used elsewhere within MCNP, affecting quantities not directly related to transport quantities. A full investigation of this was not made due to time constraints.

At this point, transport calculations were interrupted in order to correct the above quantities for a variable density atmosphere.

1. line 104: the flag NINP set equal to 0 indicating the calling subroutine is HSTORY.

2. line 105: subroutine EQDIST called to calculate the distance to next collision, PMF, corrected for a variable density atmosphere.

3. line 108: if PMF is returned with the value -1, then PMF is set equal to HUGE, a very large number used by MCNP. This effectively transports the particle out of the problem with no further collisions.

4. lines 111 thru 113: effective values of the mean free path, GS, the macroscopic cross section, QPL, and the macroscopic cross section, PLE, are recalculated. Since the exponential transform is not used in the variable density version of MCNP, there is no difference between QPL and PLE.

Transport calculations continue from this point with no further modifications.

7.2.4 Subroutine TRANSM

This subroutine calculates the transmission to the detector or nearest intersecting boundary when point detectors or ring detectors are used. Prior to calculating the transmission, subroutine DDDET is called to calculate the distance, DD, to the detector. On line 24, subroutine TRACK is called to calculate the distance to the nearest intersecting surface, DLS.

Cross sections for the cell are determined via subroutines ACETOT for neutrons and PHOTOT for photons on lines 30 and 31. Once the microscopic cross section [barns/atom] in the cell has been determined, it is multiplied by the atom density [atoms/barn-cm] in the cell to obtain the macroscopic cross section [cm^{-1}] on line 32. On line 34, the track length in the cell, D, is determined by taking the minimum of the distance to the nearest bounding surface, DLS, and the distance to the detector, DD.

Normally, the number of mean free paths to the detector or nearest intersecting boundary is computed by multiplying the track length in the cell, D, with the macroscopic cross section, PLE. At this point, normal calculations are interrupted in order to compute a new macroscopic cross section in the cell corrected for a variable density atmosphere. The following modifications were made to subroutine TR4NSM:

1. line 44: the flag NINP set equal to 1, indicating the calling subroutine is TRANSM;

2. line 45: the mean free path, XMFP [cm] is calculated by taking the reciprocal of the macroscopic cross section, PLE [cm^{-1}];

3. line 46: subroutine EQDIST is called in order to calculate a corrected mean free path, XMFP;

4. line 48: a new macroscopic cross section is determined by taking the reciprocal of the mean free path, XMFP, returned from subroutine EQDIST;

5. line 51: the corrected number of mean free paths along the track length to the detector or nearest intersecting boundary is computed using the corrected macroscopic cross section, PLE.

Calculations continue with no further modifications.

The number of mean free paths to the detector or nearest bounding surface, AMFP, is determined using the values of the macroscopic cross section in the cell, PLE, and the minimum distance to the detector or nearest bounding surface, D. The locations of the new cell and particle position are then updated.

8 Miscellaneous Information

This section presents some miscellaneous information which might be useful to future users of the modified MCNP code at AFIT. Topics discussed include: 1) the nature of the source spectra used, 2) the location of the MCNP files, and 3) comments concerning the applicability of the modified MCNP code.

8.1 Neutron-Photon Source

There were two source spectra provided with the SMAUG-II code [4]. The spectra included 37 neutron energy groups ranging from thermal energies to a maximum of 14.2 MeV and 21 photon groups ranging from 10^{-2} MeV to a maximum of 14 MeV. The neutron and photon spectrum energies and corresponding energy bin probabilities are listed in Appendix F in thesis AFIT-/GNE/ENP/91M-6. The default spectra provided with the SMAUG-II code were used in the MCNP simulations to facilitate the comparisons between SMAUG-II and MCNP. Each Monte Carlo simulation was repeated three times so primary neutrons, primary photons, and secondary photons could be generated. The source spectra were not classified, but rather generic, unclassified spectra provided with the computer code SMAUG-II.

8.2 Location of MCNP files

There are now two versions of the MCNP code version 3B at AFIT: 1) the unmodified version of MCNP and 2) the modified version of MCNP incorporating a variable density atmosphere. The location of the unmodified MCNP files and the executable program can be found on the ELXSI (GALAXY) computer at AFIT, i.e., under the following directory

```
/src/mcnp/Working
```

The location of the modified MCNP files and the executable program can be found under the following directory

```
/src/mcnp/Working1
```

The location of the cross section libraries can be found under

```
/src/mcnp/XCLib
```

Copies of both the unmodified and modified versions of MCNP were transferred to the TRITON SUN system at AFIT, in LCDR Kirk Mathews office (AFIT/ENP) under the /dculp and /dmonti directories, respectively. Once the capability to run MCNP on TRITON has been added, either MCNP version can be run. Copies of the cross section libraries were also transferred to TRITON.

The file transfer sequence to copy individual files from GALAXY to TRITON are given below

- logon to userid on GALAXY
- go to directory containing applicable files
- ftp triton
- userid
- password
- cd to target directory on triton
- mput (multiple files) or put (1 file) filename
- close
- quit

If a large number of files needs to be transferred to TRITON, use the following sequence of commands

- logon to userid on GALAXY
- go to directory containing applicable files

- ar r archive name *.*   (archives all files in directory)
- repeat ftp sequence above to transfer the archive file to triton.

Once the archive file has been transferred, perform the following commands to retrieve the archived files

- logon to userid on triton
- go to applicable directory
- ar x archive name *.*   (extracts all archived files)

8.3 Comments

8.3.1 Exponential Transform

Although modifications to incorporate a variable density atmosphere to MCNP were very successful, further work is required to investigate the effects of the exponential transform (not used thus far) on the macroscopic cross section. The exponential transform can be invoked in the HSTORY subroutine from the input file (see MCNP user's manual). Since discrete cells, for the purpose of representing the variation of air density with altitude, were not required, all transport density-dependent parameters within MCNP were first calculated based on sea-level air density. The parameters were then corrected for a variable density atmosphere. The full effects of invoking the exponential transform, which involves path length stretching, requires investigation before this feature can be used with confidence.

8.3.2 Other Density-Dependent Features

There are other density-dependent features used by MCNP which may require modification. These include: 1) fission heating, 2) dose-response relationships, 3) cell energy deposition, and 4) track mean free path. Fission heating (if this were somehow invoked in the atmosphere), cell energy deposition, and dose calculations all involve knowing either the air density and the density of some other material (if applicable) within the atmosphere. Modifications would be required to enable the user to use the variable density version of MCNP with more than one material. The track mean free path is printed to the output listing upon program completion. It is computed based on the average mean free path along all particle tracks within a cell. Again, fewer discrete cells are used in the modified MCNP code, and thus the track mean free path within a cell becomes meaningless. This will not affect final results; the track mean free path is used only as diagnostic information.

References

1. Monti, David L. Capt, USAF. High Altitude Neutral Particle Transport Using the Monte Carlo Simulation Code MCNP with Variable Density Atmosphere, MS Thesis, unpublished, Air Force Institute of Technology (AU), AFIT/GNE-/ENP/91M-6, March 1991.

2. RSIC Computer Code Collection CCC-200. "MCNP Version 3B, Monte Carlo Neutron and Photon Transport System", Contributed by Los Alamos National Laboratory. March 1989.

3. U.S. Standard Atmosphere, 1976. Washington, D.C.: U.S. Government Printing Office, October 1976.

4. Murphy, Harry M. "A User's Guide to the SMAUG-II Computer Code", Air Force Weapons Laboratory, Kirtland AFB, NM. 4 March 1981.

Appendix A: User-Written Subroutines and Functions

```
      SUBROUTINE EQDIST(EDST, DTD, ZO, WO, NINP)

*=================================================================
*
*    THIS SUBROUTINE CALCULATES 2 PARAMETERS, DEPENDING ON
*    INPUT:
*
*      1) IF THE INPUT VALUE IS THE SAMPLED DISTANCE TO
*         COLLISION FROM THE 'HSTORY.F' SUBROUTINE, THIS
*         SUBROUTINE RETURNS AN EQUIVALENT DISTANCE TO
*         COLLISION BASED ON A VARIABLE DENSITY ATMOSPHERE.
*      2) IF THE INPUT VALUE IS THE CALCULATED MEAN FREE PATH
*         FROM THE 'TRANSM.F' SUBROUTINE, THIS SUBROUTINE
*         RETURNS AN EQUIVALENT MEAN FREE PATH BASED ON A
*         VARIABLE DENSITY ATMOSPHERE.
*
*    ALL UNITS ARE IN cgs, CONSISTENT WITH MCNP.
*
*=================================================================

*=================================================================
*
*    THE FOLLOWING IS A LIST OF ARRAYS AND VARIABLES USED IN
*    THIS SUBROUTINE AND EXTERNAL TO THE FUNCTIONS.
*
*=================================================================
*
*     ZO      - CURRENT PARTICLE ALTITUDE
*     WO      - Z-DIRECTION COSINE RELATIVE TO POLAR ANGLE
*     EDST    - EQUIVALENT DISTANCE TO COLLISION OR
*             - EQUIVALENT MEAN FREE PATH TO BOUNDARY/DETECTOR
*     DTD     - DISTANCE TO BOUNDARY OR DETECTOR
*     NINP    - FLAG INDICATING WHICH CALCULATIONS ARE
*               PERFORMED
*     NR1     - NUMBER OF SCALE HEIGHT REGIONS
*
*=================================================================
      include 'CM.inc'
      EXTERNAL MASI,DNTY,ZMAS,DMI,MIZ,EQUIVDST
      DOUBLE PRECISION ZO,ZD,WO,MICA,DENCA,DENCP,ZC
      DOUBLE PRECISION MASI,DNTY,ZMAS,DMI,MIZ,EQUIVDST,DTD
      DOUBLE PRECISION EDST,MICP,MIDA,MIINF,MIPD,MIPDH

C  DEFINE MASS INTEGRAL AT INFINITY [g/cm²] (ZO > 990 KM)
      MIINF = 1035.635131402448

C  DETERMINE OPTIMUM ARRAY SEARCH PARAMETERS
      L=NR1/2
      IF(WO.GE.O..AND.ZO.LT.ZI(L))THEN
         I1=1
```

```
            I2=NR1
            I3=1
         ELSEIF(WO.GE.O..AND.ZO.GE.ZI(L))THEN
            I1=L
            I2=NR1
            I3=1
         ELSEIF(WO.LT.O..AND.ZO.LT.ZI(L))THEN
            I1=L
            I2=1
            I3=-1

         ELSEIF(WO.LT.O..AND.ZO.GE.ZI(L))THEN
            I1=NR1
            I2=1
            I3=-1
         ENDIF

         IF(NINP.EQ.O)THEN

*===========================================================================
C  PERFORM CALCULATIONS BASED ON INPUTS FROM 'HSTORY.F'
C    SUBROUTINE.
*===========================================================================

C  CALCULATE MASS INTEGRAL ALONG PARTICLE DIRECTION IN A
C    HOMOGENEOUS SEA-LEVEL ATMOSPHERE.
            MIPDH = DMI(EDST)

C  CALCULATE MASS INTEGRAL AT CURRENT PARTICLE ALTITUDE IN
C    A VARIABLE DENSITY ATMOSPHERE.
            MICA = MASI(ZO,I1,I2,I3)

C  CALCULATE REQUIRED MASS INTEGRAL AT COLLISION ALTITUDE IN
C     A VARIABLE DENSITY ATMOSPHERE.
            MICP = MIZ(MICA, MIPDH, WO)

C  CHECK TO SEE IF COLLISION ALTITUDE IS WITHIN PROBLEM
C     LIMITS.
            IF (MICP.GE.O..AND.MICP.LE.MIINF) THEN

C  CALCULATE COLLISION ALTITUDE GIVEN REQUIRED MASS INTEGRAL
C     ALONG PARTICLE PATH.
            ZC = ZMAS(MICP,I1,I2,I3)

C  CALCULATE DENSITY AT CURRENT PARTICLE ALTITUDE AND AT
C     COLLISION ALTITUDE FOR CASES WHERE WO IS APPROXIMATELY
C     EQUAL TO O.
               IF((ZC-ZO).LT.1000.) THEN
                  DENCA = DNTY(ZO,I1,I2,I3)
                  DENCP = DNTY(ZC,I1,I2,I3)
               ENDIF
```

```
C  CALCULATE NEW EQUIVALENT DISTANCE TO COLLISION.
            EDST = EQUIVDST(ZC, ZO, WO, MIPDH, DENCA, DENCP)
         ELSE
            EDST = -1.
         ENDIF

      ELSEIF(NINP.EQ.1)THEN

*=====================================================================
C  PERFORM CALCULATIONS BASED ON INPUTS FROM 'TRANSM.F'
C     SUBROUTINE.
*=====================================================================

C  CALCULATE ALTITUDE AT DETECTOR OR BOUNDARY CROSSING.
         ZD = ZO + WO*DTD
         IF(ZD.LT.O..OR.ZD.GT.ZI(NR2))THEN
            WRITE(IUO,'(15X,A38)')MSG
            EDST=-1
            RETURN
         ENDIF

C  CALCULATE MASS-INTEGRAL ALONG PARTICLE DIRECTION IN
C    A HOMOGENEOUS ATMOSPHERE.
C    SEA-LEVEL ATMOSPHERE.
         MIPDH = DMI(DTD)

C  CALCULATE MASS INTEGRAL AT CURRENT PARTICLE ALTITUDE IN
C    A VARIABLE DENSITY ATMOSPHERE.
         MICA = MASI(ZO,I1,I2,I3)

C  CALCULATE MASS INTEGRAL AT DETECTOR/BOUNDARY ALTITUDE IN
C    A VARIABLE DENSITY ATMOSPHERE.
         MIDA = MASI(ZD,I1,I2,I3)

C  CALCULATE MASS INTEGRAL ALONG PARTICLE DIRECTION IN
C    A VARIABLE DENSITY ATMOSPHERE.
         IF (ABS(ZD-ZO).GE.1000) THEN
            MIPD = (MIDA-MICA)/WO
         ELSE
            DENCA = DNTY(ZO,I1,I2,I3)
            DENCP = DNTY(ZD,I1,I2,I3)
            MIPD = DTD*(DENCA+DENCP)/2.
         ENDIF

C  CALCULATE A NEW MEAN FREE PATH BASED ON A VARIABLE
C    DENSITY ATMOSPHERE.
         EDST = MIPDH * EDST / MIPD

      ENDIF
      RETURN
      END
```

```
   FUNCTION EQUIVDST(ZN, Z, W, MIPD, DCA, DCP)

*================================================================
*
C  THIS FUNCTION CALCULATES A NEW EQUIVALENT DISTANCE TO
C     COLLISION ALONG THE PARTICLE DIRECTION WHICH GIVES THE
C     SAME MASS INTEGRAL AS FOUND IN A HOMOGENEOUS SEA-LEVEL
C     ATMOSPHERE.
*
*================================================================

*================================================================
*
C      VARIABLES USED IN THIS FUNCTION:
C
C           ZN        - NEW ALTITUDE IN A VARIABLE DENSITY
C                         ATMOSPHERE
C           Z         - CURRENT PARTICLE ALTITUDE
C           W         - Z-DIRECTION COSINE RELATIVE TO THE
C                         POLAR ANGLE
C           DCA       - DENSITY AT CURRENT PARTICLE ALTITUDE
C           DCP       - DENSITY AT NEW ALTITUDE
C           MIPD      - MASS INTEGRAL ALONG PARTICLE DIRECTION
C                         (HOMOGENEOUS SEA-LEVEL ATMOSPHERE)
C           EQUIVDST - EQUIVALENT DISTANCE TO COLLISION OR
C                         MEAN FREE PATH TO BOUNDARY/DETECTOR
*
*================================================================
      DOUBLE PRECISION ZN,Z,W,MIPD,DCA,DCP,EQUIVDST
C
C  IF THE DIFFERENCE IN ALTITUDES BECOMES SMALL ( < 10m ),
C    USE THE AVERAGE DENSITY BETWEEN THE NEW ALTITUDE AND
C    CURRENT ALTITUDE.
      IF (ABS(ZN-Z).GE.1000) THEN
         EQUIVDST = (ZN-Z)/W
      ELSE
         EQUIVDST = MIPD / (0.5*(DCA+DCP))
      ENDIF
      END


   FUNCTION MASI(Z,I1,I2,I3)

*================================================================
*
C      GIVEN AN ALTITUDE, FIND THE MASS INTEGRAL.
C
C      VARIABLES USED IN THIS FUNCTION:
C
C         Z     - GIVEN PARTICLE ALTITUDE
C         ZI    - ARRAY CONTAINING BASE ALTITUDE OF REGIONS
C         DENI  - ARRAY CONTAINING DENSITY AT ZI
```

```
C        AMII - ARRAY CONTAINING MASS INTEGRAL AT ZI
C        SHFM - ARRAY CONTAINING MASS INTEGRAL SCALE HEIGHT
C               FACTORS
C        MASI - MASS INTEGRAL AT GIVEN ALTITUDE
*
*==========================================================
      include 'CM.inc'
      DOUBLE PRECISION MASI,Z
      DO 10 K=I1,I2,I3
         IF (Z.GE.ZI(K).AND.Z.LE.ZI(K+1)) THEN
           MASI =
AMII(K)+DENI(K)*SHFM(K)*(1-EXP(-(Z-ZI(K))/SHFM(K)))
         END IF
 10   CONTINUE
      END


   FUNCTION ZMAS(MI,I1,I2,I3)

*==========================================================
*
C     GIVEN A MASS INTEGRAL, FIND THE ALTITUDE.
C
C     VARIABLES USED IN THIS FUNCTION:
C
C        MI   - GIVEN MASS INTEGRAL
C        SHFM - ARRAY CONTAINING MASS INTEGRAL SCALE HEIGHT
C               FACTORS
C        ZI   - ARRAY CONTAINING BASE ALTITUDE OF REGIONS
C        AMII - ARRAY CONTAINING MASS INTEGRAL AT ZI
C        DENI - ARRAY CONTAINING DENSITY AT ZI
C        ZMAS - ALTITUDE CORRESPONDING TO GIVEN MASS INTEGRAL
*
*==========================================================
      include 'CM.inc'
      DOUBLE PRECISION MI,ARG,ZMAS
      DO 30 K=I1,I2,I3
         IF (MI.GE.AMII(K).AND.MI.LE.AMII(K+1)) THEN
            ARG = LOG(1 + (AMII(K) - MI) / (DENI(K) *
SHFM(K)))
            ZMAS = ZI(K) - SHFM(K) * ARG
         END IF
 30   CONTINUE
      END
      FUNCTION DNTY(Z,I1,I2,I3)

*=====================================================================
*
C     GIVEN AN ALTITUDE, FIND THE DENSITY.
C
C     VARIABLES USED IN THIS FUNCTION:
C
C          Z    - GIVEN PARTICLE ALTITUDE
C          ZI   - ARRAY CONTAINING BASE ALTITUDE OF REGIONS
C          DENI - ARRAY CONTAINING DENSITY AT ZI
C          SHFD - ARRAY CONTAINING DENSITY SCALE HEIGHT
C                 FACTORS
C          DNTY - DENSITY CORRESPONDING TO GIVEN ALTITUDE
*
*=====================================================================
      include 'CM.inc'
      DOUBLE PRECISION Z,DNTY
      DO 10 K=I1,I2,I3
         IF (Z.GE.ZI(K).AND.Z.LE.ZI(K+1)) THEN
            DNTY = DENI(K) * EXP(-(Z-ZI(K)) / SHFD(K))
         END IF
 10   CONTINUE
      END

      FUNCTION DMI(DIST)

*=====================================================================
*
C     CALCULATE MASS INTEGRAL BETWEEN TWO POINTS IN A
C     SEA-LEVEL HOMOGENEOUS ATMOSPHERE.
C
C     VARIABLES USED IN THIS FUNCTION:
C
C          DIST - DISTANCE TO COLLISION OR MEAN FREE PATH TO
C                 BOUNDARY/DETECTOR
C          DMI  - MASS INTEGRAL ALONG PARTICLE DIRECTION
*
*=====================================================================
      DOUBLE PRECISION DIST,DMI
C
C  CALCULATE MASS INTEGRAL ALONG PARTICLE PATH.
         DMI = 1.225E-3*DIST
      END
      FUNCTION MIZ(MIP, DMI, W)

*===================================================================
*
C     CALCULATE MASS INTEGRAL AT NEW ALTITUDE IN AN EXPONENTIAL
C       ATMOSPHERE.
C
C        VARIABLES USED IN THIS FUNCTION:
C
C             MIP - MASS INTEGRAL AT CURRENT PARTICLE ALTITUDE
C                   IN A VARIABLE DENSITY ATMOSPHERE
C             DMI - MASS INTEGRAL ALONG PARTICLE DIRECTION IN A
C                   HOMOGENEOUS SEA-LEVEL ATMOSPHERE
C             W   - Z-DIRECTION COSINE RELATIVE TO THE POLAR
C                   ANGLE
C             MIZ - MASS INTEGRAL AT NEW ALTITUDE IN A
C                   VARIABLE DENSITY ATMOSPHERE
*
*===================================================================
         DOUBLE PRECISION MIP,DMI,W,MIZ
C
C     CALCULATE MASS INTEGRAL AT NEW ALTITUDE.
         MIZ = W*DMI + MIP
         END


      BLOCK DATA ATINIT

*===================================================================
*
C        THIS DATA BLOCK IS USED TO SUPPORT THE VARIABLE
C        DENSITY ATMOSPHERIC MODEL.  INITIALIZE COMMON /ATMCOM/
C        WITH VALUES OF ATMOSPHERIC PARAMETERS.
*
*===================================================================
         include 'CM.inc'
*===================================================================
*
C           SHFM - ARRAY CONTAINING MASS INTEGRAL SCALE HEIGHT
C                  FACTORS
C           SHFD - ARRAY CONTAINING DENSITY SCALE HEIGHT FACTORS
C           ZI   - ARRAY CONTAINING BASE ALTITUDE OF VERTICAL
C                  REGIONS
C           DENI - ARRAY CONTAINING DENSITY AT ZI
C           AMII - ARRAY CONTAINING MASS INTEGRAL AT ZI
*
*===================================================================
*
         DATA SHFM/1034124.022372781,1009981.924149543,
        1          986693.172619466,964503.5327215038,
        2          941155.9145785341,888092.312585042,
        3          692868.8443723,637442.9797604,
     4          626192.9959628,639736.90617414,
     5          47557.78050254,710362.4275112881,
     6          838211.2645959415,776503.932153373,
     7          683277.3511724839,612608.5454716644,
     8          557270.926150725,568864.8492265636,
     9          619813.9513853799,935109.6091771976,
     +          1267217.415038409,1578526.59297959,
     +          1870150.85627943,2219020.933057442,
     +          2714282.134795458,3257224.61137614,
     +          3991584.358044663,4584848.61083246,
     +          5162659.761537237,5791446.057543592,
     +          5501266.555948576,6836368.561027177,
     +          7834040.063075421,9814372.605576273,
     +          13917805.4965649,18877525.61508398/
      DATA SHFD/1030391.726068488,1006927.055016296,
     1          983153.5607496358,960534.1411264988,
     2          937232.106253662,866352.1969023093,
     3          664086.7633898156,637638.46529,
     4          627630.265119,642604.7540969,
     5          654589.3995756,736011.6057506185,
     6          834195.106975831,758287.3616637037,
     7          666098.102745308,592758.5281268486,
     8          553227.4697274631,570413.0947863342,
     9          678176.4068898259,997277.941535646,
     +          1324262.260093013,1632165.675462571,
     +          1919412.5389299,2313392.496297635,
     +          2797412.470495623,3408285.286706647,
     +          4112545.885579616,4680457.552711085,
     +          5311463.853509105,5870652.50294756,
     +          6360111.158318114,6989325.564711994,
     +          8162740.323944079,10419949.4345468,
     +          14188168.29614109,19869924.24084625/
      DATA ZI/0.,1.D+5,2.D+5,3.D+5,4.D+5,5.D+5,1.D+6,
     1        1.5D+6,2.D+6,2.5D+6,3.D+6,4.D+6,5.D+6,
     2        6.D+6,7.D+6,8.D+6,9.D+6,1.D+7,1.1D+7,
     3        1.2D+7,1.3D+7,1.4D+7,1.5D+7,1.6D+7,1.8D+7,
     4        2.0D+7,2.4D+7,2.8D+7,3.2D+7,4.0D+7,4.8D+7,
     5        5.6D+7,6.4D+7,7.2D+7,8.0D+7,8.8D+7,9.9D+7/
      DATA DENI/1.225D-3,1.1117D-3,1.0066D-3,9.0925D-4,
     1         8.1935D-4,7.3643D-4,4.1351D-4,1.9476D-4,
     2         8.891D-5,4.0084D-5,1.841D-5,3.9957D-6,
     3         1.0269D-6,3.0968D-7,8.2829D-8,1.8458D-8,
     4         3.416D-9,5.604D-10,9.708D-11,2.222D-11,
     5         8.152D-12,3.831D-12,2.076D-12,1.233D-12,
     6         5.194D-13,2.541D-13,7.858D-14,2.971D-14,
     7         1.264D-14,2.803D-15,7.208D-16,2.049D-16,
     8         6.523D-17,2.448D-17,1.136D-14,6.464D-18,
     9         3.716D-18/
      DATA AMII/0.,116.7635,222.607167,318.334333,
     1         404.704533,482.4368,763.994467,911.2723,
     2         978.7592666666669,1009.379620,
     3         1023.286286666667,1032.662946666667,
     4          1034.806793333334,1035.406480000001,
     5          1035.580609766667,1035.624107866667,
     6          1035.633205146667,1035.634792366667,
     7          1035.635056196001,1035.635104380667,
     8          1035.635118027400,1035.635123665300,
     9          1035.635126503134,1035.635128111100,
     +          1035.635129736200,1035.635130471237,
     +          1035.635131056504,1035.635131255017,
     +          1035.635131334304,1035.635131385704,
     +          1035.635131397859,1035.635131400898,
     +          1035.635131401864,1035.635131402191,
     +          1035.635131402325,1035.635131402394,
     +          1035.635131402448/
C
      END
```

Appendix B: Modified MCNP Subroutines and Files

```
C     INCLUDE FILE ZC.inc

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C        CODE NAME AND VERSION NUMBER.
      CHARACTER KOD*8,VER*5
      PARAMETER (KOD='MCNP',VER='3B3')
      PARAMETER (MSG='DETECTOR ALTITUDE EXCEEDS MODEL
LIMITS')
      PARAMETER (MSG1='*** NOTE: USING VARIABLE DENSITY MODEL
***')
C
C        PROCESSOR-DEPENDENT NAMED CONSTANTS.
C        MDAS IS THE INITIAL SIZE OF DYNAMICALLY ALLOCATED
COMMON     C        /DAC/ ON SYSTEMS WHERE MEMORY
ADJUSTMENT IS NOT AVAILABLE,
C        SET MDAS LARGE ENOUGH FOR YOUR BIGGEST PROBLEM.
      PARAMETER (MDAS=500000)
      PARAMETER (NDP2=2,HUGE=1D37)
      PARAMETER (FTLS=.2d0,DFTINT=100.d0)
C
C        ARRAY DIMENSIONS.  I/O UNIT NUMBERS.  GENERAL
CONSTANTS.
      PARAMETER
(MAXE=50,MAXF=16,MAXV=18,MAXW=2,MEMAX=150,MINK=200,
     1
MIPT=2,MJSF=9,MKFT=7,MKTC=22,MSEB=301,MXC=180,MXDT=20,MXDX=5,
     2
MXLV=10,MXMTX=100,NBMX=100,NDEF=14,NOVR=5,IUI=1,IUO=2,IUR=3,
     3 IUX=4,IUD=7,IUB=8,IUP=9,IUS=10,IU1=11,IU2=12,IUSW=13,
     4 IUSR=14,IUSC=15,IUC=16,IUT=17,IUZ=18,IUK=19,JTTY=6,
     5
ZERO=0.d0,ONE=1.d0,PIE=3.1415926535898d0,FIVE19=(ZERO+5.0d0)*
*19,
     6 AVGDN=.59703109d0,GELEC=.511008d0,GNEUT=939.58d0,
     7 SLITE=299.7925d0,NR1=36,NR2=37)
C
C
C-------------------------------------------------------------
C
```

```
C     INCLUDE FILE CM.inc


 include 'ZC.inc'
      DOUBLE PRECISION
SHFM(NR),SHFD(NR),ZI(NR),DENI(NR),AMII(NR)
C
C    **********************  STATIC COMMON
*************************
C
C         FIXED COMMON -- CONSTANT AFTER THE PROBLEM IS
INITIATED.
      COMMON /FIXCOM/
ATWT(MEMAX),BCW(2,3),DDG(MIPT,MXDT),DXW(MIPT,3),
     1
DXX(MIPT,5,MXDX),ECF(MIPT),EMCF(MIPT),EMX,ERGSAB(0:MAXE),
     2
ESPL(MIPT,10),FME(MEMAX),FNW,RIM,RSSP,SNIT,SRV(3,MAXV),
     2
TBLTMP(MAXE),TCO(MIPT),THGF(0:50),WC1(MIPT),WC2(MIPT),WCS1(MI
PT),
     3 WCS2(MIPT),WWG(7),WWP(MIPT,5),
     3 ZFIXCM,
     4
ICW,IDEFV(MAXV),IDRC(MXDT),IFFT,IGM,IGWW,IKZ,IMG,IMT,INK(MINK
),
     4
IPLT,IPTY,ISB,ISSW,IVDD(MAXF),IVDIS(MAXV),IVORD(MAXF),JGM(MIP
T),
     5
JTLX,JUNF,JXS(32,MAXE),KFL,KNODS,KNRM,KPT(MIPT),KUFIL(2,6),
     6
KXS(MAXE),LDR,LFCDG,LFCDJ,LME(MIPT,MEMAX),LMT(MEMAX),LNP,
     6
LOCDT(2,MXDT),LVCDG,LVCDJ,LXS,MBNK,MCAL,MCT,MGEGBT(MIPT),MLAJ
,
     7
MLJA,MODE,MRL,MSD,MSRK,MXA,MXAFM,MXAFS,MXE,MXF,MXFO,MXF2,MXF3
,
     8
MXJ,MXT,MXTR,NDET(MIPT),NDTT,NDX(MIPT),NGWW(MIPT),NISS,NJSR,N
JSS,
     8
NKXS,NLEV,NLJA,NNPOS,NORD,NP1,NPIKMT,NPN,NRCD,NRSS,NSPH,NSR,
     9 NSRCK,NTAL,NTY(MAXE),NVEC,NWW(MIPT),NXS(16,MAXE)
C
C         OFFSETS FOR VIRTUAL ARRAYS IN DYNAMICALLY ALLOCATED
STORAGE.
      COMMON /FIXCOM/
LARA,LCMG(MIPT),LDEN,LDXP,LEAA,LEWG(MIPT),LFIM,
     1
LFMG,LFOR,LFRC,LGMG(MIPT),LGVL(MIPT),LGWT,LPMG(MIPT),LQAX,LRH
```

```
O,
     2
LSCF,LSMG(MIPT),LSPF,LSQQ,LSSO,LTDS,LTMP,LTRF,LTTH,LVCL,LVEC,
     3
LVOL,LWWE(MIPT),LWWF(MIPT),LIPA,LIPT,LISS,LITD,LJAR,LJPT,LJSC
,
     4
LJSS,LJTF,LJUN,LJVC,LKCP,LKSD,LKST,LLAF,LLAT,LLCA,LLFC,LLFT,L
LJA,
     5
LLCT,LLST,LLSC,LMAT,LMFL,LMLL,LNCL,LNSF,LDDM,LDDN,LDEC,LDXC,L
DXD,
     6
LFLX(MIPT),LFSO,LGWW(MIPT),LPAC,LPAN,LPCC,LPWB,LRKP,LTFC,LWNS
,
     7
LISE,LJFQ,LLAJ,LLCJ,LLSE,LNPW,LNSL,LNTB,LSCR,LDRC,LFDD,LGNR,L
PIK,
     8 LIFL,LIGM,LPC2,LJFL,LJFT,LTAL,LBNK,LXSS,
     9 MFIXCM
C
C         THIS IS A USER-DEFINED COMMON BLOCK CONTAINING
PARAMETER ARRAYS
C         FOR THE VARIABLE DENSITY ATMOSPHERIC MODEL.
      COMMON /ATMCOM/
SHFM(NR1),SHFD(NR1),ZI(NR2),DENI(NR2),AMII(NR2)
C
      PARAMETER
(NFIXCM=MIPT*MXDT+2*MEMAX+5*MIPT*MXDX+3*MAXV+2*MAXE+
     1 25*MIPT+71,LFIXCM=
     2
3*MXDT+MINK+50*MAXE+(1+MIPT)*MEMAX+17*MIPT+2*MAXV+2*MAXF+159)
      DIMENSION GFIXCM(NFIXCM),JFIXCM(LFIXCM)
      EQUIVALENCE (ATWT,GFIXCM),(ICW,JFIXCM)
C
C         VARIABLE COMMON -- VARIABLE BUT REQUIRED FOR A
CONTINUE RUN.
C         ARRAYS THAT ARE BACKED UP WHEN A TRACK IS LOST.
      COMMON /VARCOM/
FEBL(2,16),PAX(MIPT,6,16),RDUM(50),RLT(2),SMUL(3),
     1 SUMK(3),TMAV(MIPT,3),TWAC,TWSS,WSSI(7),
     2 ZVARCM,
     3
IDUM(50),IST,IXAK,JRAD,KCSF,NBHWM,NBT(MIPT),NBY,NCT(MIPT),
     4
NDRR(MAXE),NETB(2),NPS,NQSW,NRSW,NSA,NSS,NSSI(8),NTSS,NWSB,NW
SE,
     5 NWSG,NWSL,NWST,
     6 MVARCM

      PARAMETER (NVARCM=99*MIPT+100,LVARCM=MAXE+2*MIPT+78)
      DIMENSION GVARCM(NVARCM),JVARCM(LVARCM)
      EQUIVALENCE (FEBL,GVARCM),(IDUM,JVARCM)
C
C         NOT-BACKED-UP VARIABLE COMMON
      COMMON /NBVCOM/
CPK,CTS,DBCN(20),DDX(MIPT,2,MXDX),DMP,FIS,OSUM(3),
     1
OSUM2(3,3),PRN,RANI,RANJ,RIJK,RKK,RSUM(2),RSUM2(2,2),STLS,
     2 WGTS(2),WTO,
     3 ZNBVCM,
     4
KCT,KCY,KNOD,KTLS,KZKF,LOST(2),NBOV,NERR,NFER,NLAJ,NLSE,NPC(2
0),
     5
NPD,NPNM,NPP,NPPM,NPSR,NQSS,NRN,NRRS,NSKK,NTC,NTC1,NWER,
     6 NWWS(2,99),NZIP,NZIX,NZIY(MXDX,MIPT),
     7 MNBVCM
      PARAMETER (NNBVCM=2*MIPT*MXDX+52,LNBVCM=MIPT*MXDX+245)
      DIMENSION GNBVCM(NNBVCM),JNBVCM(LNBVCM)
      EQUIVALENCE (CPK,GNBVCM),(KCT,JNBVCM)
C
C         EPHEMERAL COMMON -- NOT NEEDED AFTER THE CURRENT
RUN.
      COMMON /EPHCOM/
ANG(3),CPA,CTME,RANB,RANK,RANL,RANS,SSB(10),
     1
TPP(20),UDT(10,0:MXLV),UDTS(10*(1+MXLV)),WNVP(4),XHOM,YHOM,
     2 ZEPHCM,
     3
ICHAN,ICS,IDMP,IFILE,ILN,ILN1,IMTX(MXMTX),INDT,INFORM,IOVR,IT
AL,
     4
ITERM,ITFXS,ITOTNU,ITTY,IUOU,JCHAR,JFCN,JGF,JGXA(2),JGXO(2),
     5
JOVR(NOVR),JVP,KONRUN,KPROD,LDQ,LFATL,LFLL,LGC(101),LSPEED,MI
X,
     6 MNK,NBNK,NCH(MIPT),NDE,NKRP,NST,
     7 MEPHCM
      PARAMETER
(NEPHCM=66+20*MXLV,LEPHCM=MXMTX+NOVR+MIPT+137)
      DIMENSION GEPHCM(NEPHCM),JEPHCM(LEPHCM)
      EQUIVALENCE (ANG,GEPHCM),(ICHAN,JEPHCM)
C
C         PBL COMMON -- PARTICLE AND COLLISION DESCRIPTORS.
C         IF /PBLCOM/ IS MODIFIED, /PB9COM/ MUST BE CHANGED TO
MATCH IT.
      COMMON /PBLCOM/
XXX,YYY,ZZZ,UUU,VVV,WWW,ERG,WGT,TME,VEL,ICL,JSU,
     1
IPT,NPA,IEX,NODE,IDX,NCP,KRN,JGP,DLS,DXL,DTC,FIML,FIM1,FISMG,
     2 WTFASV,LEV,KKBNK,III,JJJ,KKK,IAP,SPARE1,SPARE2,SPARE3,
     3 MPBLCM
      PARAMETER (LPBLCM=NDP2*20+17)
      DIMENSION JPBLCM(LPBLCM),PBL(10)
      EQUIVALENCE (XXX,JPBLCM,PBL)
C
      COMMON /TABLES/ EBL(16),TALB(8,2),JSF(MJSF),NVS(MAXV)
C
C        CHARACTER COMMON -- CHARACTER VARIABLES AND ARRAYS.
      CHARACTER
AID*80,AID1*80,AIDS*80,CHCD*10,EXMS*80,HBLN(MAXV,2)*3,
     1
HBLW(MAXW)*3,HCS(2)*7,HFT(MKFT)*8,HFU(2)*11,HMM(MEMAX)*10,
     2
HMT(MXMTX)*10,HNP(MIPT)*7,HOVR*8,HSD(2)*10,IBIN*8,IDTM*19,
     3
IDTMS*19,ILBL(8)*8,KLIN*80,KODS*8,KOVR(NOVR)*6,KSF(29)*3,
     4
LODDAT*8,LODS*8,MSUB(NDEF)*10,PROBID*19,PROBS*19,RFQ(10)*57,
     5 UFIL(3,6)*11,VERS*5,XDATE(MAXE)*8,XLIST(MAXE)*10,
     6 XSCRD(MAXE)*(MXC)
      COMMON /CHARCM/
AID,AID1,AIDS,CHCD,EXMS,HBLN,HBLW,HCS,HFT,HFU,HMM,
     1
HMT,HNP,HOVR,HSD,IBIN,IDTM,IDTMS,ILBL,KLIN,KODS,KOVR,KSF,LODD
AT,
     2 LODS,MSUB,PROBID,PROBS,RFQ,UFIL,VERS,XDATE,XLIST,XSCRD

C        ISUB:  NAMES OF FILES
      CHARACTER*8
INP,OUTP,RUNTPE,SRCTP,XSDIR,PIX,WSSA,RSSA,COM,COMOUT,
     1 PLOTM,MCTAL,DUMN1,DUMN2,ISUB(NDEF)
      COMMON /CHARCM/
INP,OUTP,RUNTPE,SRCTP,XSDIR,PIX,WSSA,RSSA,COM,
     1 COMOUT,PLOTM,MCTAL,DUMN1,DUMN2
      EQUIVALENCE (ISUB,INP)
C
      COMMON /PB9COM/
XXX9,YYY9,ZZZ9,UUU9,VVV9,WWW9,ERG9,WGT9,TME9,VEL9,
     1
ICL9,JSU9,IPT9,NPA9,IEX9,NODE9,IDX9,NCP9,KRN9,JGP9,DLS9,DXL9,
     2
DTC9,FIML9,FIM19,FISMG9,WTFAS9,LEV9,KKBNK9,III9,JJJ9,KKK9,IAP
9,
     3 SP7,SP8,SP9,
     4 MPB9CM
      PARAMETER (LPB9CM=LPBLCM)
      DIMENSION JPB9CM(LPB9CM)
      EQUIVALENCE (XXX9,JPB9CM)
C
      COMMON /PB8COM/
XXX8,YYY8,ZZZ8,UUU8,VVV8,WWW8,ERG8,WGT8,TME8,VEL8,
     1 ICL8,JSU8,IPT8
      PARAMETER (LPB8CM=NDP2*10+3)
      DIMENSION JPB8CM(LPB8CM)
      EQUIVALENCE (XXX8,JPB8CM)
C
      COMMON /BACKUP/ GVBU(NVARCM),JVBU(LVARCM)
C
C    *****************  DYNAMICALLY ALLOCATED COMMON
*****************
C
      COMMON /DAC/ DAS(MDAS/NDP2)
C
C        FIXED DYNAMICALLY ALLOCATED COMMON.
      DIMENSION
AAAFD(2),ARA(1),CMG(1),DEN(1),DXCP(MIPT,1),EAA(1),
     1
EWWG(1),FIM(MIPT,1),FMG(1),FOR(MIPT,1),FRC(1),GMG(1),GVL(1),
     2
GWT(1),PMG(1),QAX(MIPT,1),RHO(1),SCF(1),SMG(1),SPF(4,2),
     3
SQQ(12,1),SSO(1),TDS(1),TMP(1),TRF(17,0:0),TTH(1),VCL(3,7,1),
     4 VEC(3,1),VOL(1),WWE(1),WWF(1),
     5
IIIFD(1),IPAN(1),IPTAL(8,5,1),ISS(1),ITDS(1),JASR(1),JPTAL(8,
1),
     6
JSCN(1),JSS(1),JTF(8,1),JUN(1),JVC(1),KCP(1),KSD(21,1),KST(1)
,
     7 LAF(3,3),LAT(2,1),LCA(1),LFCL(1),LFT(MKFT,1),LJA(1),
     8
LOCCT(MIPT,1),LOCST(MIPT,1),LSC(1),MAT(1),MFL(3,1),MLL(2,1),
     9 NCL(1),NSF(1)
      EQUIVALENCE
(DAS,AAAFD,ARA,CMG,DEN,DXCP,EAA,EWWG,FIM,FMG,FOR,FRC,
     1
GMG,GVL,GWT,PMG,QAX,RHO,SCF,SMG,SPF,SQQ,SSO,TDS,TMP,TRF,TTH,V
CL,
     2
VEC,VOL,WWE,WWF,IIIFD,IPAN,IPTAL,ISS,ITDS,JASR,JPTAL,JSCN,JSS
,
     3
JTF,JUN,JVC,KCP,KSD,KST,LAF,LAT,LCA,LFCL,LFT,LJA,LOCCT,LOCST,
LSC,
     4 MAT,MFL,MLL,NCL,NSF)
C
C        VARIABLE DYNAMICALLY ALLOCATED COMMON.
      DIMENSION
AAAVD(1),DDM(2,1),DDN(23,1),DEC(2,1),DXC(2,1),
     1
DXD(MIPT,23,MXDX),FLX(1),FSO(1),GWW(2,9,1),PAC(MIPT,10,1),
     2
PAN(MIPT,6,1),PCC(3,1),PWB(MIPT,16,1),RKPL(5,1),TFC(3,20,1),
     3 WNS(2,30),
     4
IIIVD(1),ISEF(2,1),JFQ(8,0:1),LAJ(1),LCAJ(1),LSE(1),NPSW(1),
     5 NSL(10,1),NTBB(5,1)
      EQUIVALENCE
(DAS,AAAVD,DDM,DDN,DEC,DXC,DXD,FLX,FSO,GWW,PAC,PAN,
     1
PCC,PWB,RKPL,TFC,WNS,IIIVD,ISEF,JFQ,LAJ,LCAJ,LSE,NPSW,NSL,NTB
B)
C         EPHEMERAL DYNAMICALLY ALLOCATED COMMON.
      DIMENSION
SCR(1),DRC(16,1),FDD(2,1),GENR(1),PIK(1),IFL(1),
     1 IGMSAV(1),IPAC2(1),JFL(1),JFT(1)
      EQUIVALENCE
(DAS,SCR,DRC,FDD,GENR,PIK,IFL,IGMSAV,IPAC2,JFL,JFT)
C
C         TALLIES, BANK, AND CROSS-SECTIONS IN DAC.
      DIMENSION TAL(1),IBNK(1),XSS(1)
      EQUIVALENCE (DAS,TAL,IBNK,XSS)
C
C         CROSS SECTIONS ARE REAL ON ALL KINDS OF COMPUTERS.
      REAL XSS
      REAL YSS(1)
      EQUIVALENCE (YSS,XSS)
C
C         DYNAMICALLY ALLOCATED COMMON FOR THE IMCN OVERLAY.
      DIMENSION JTR(1),AWT(1),BBV(1),PRB(1),RTP(1),SFB(1),
     1
IPNT(2,MKTC,0:1),JASW(1),KAW(1),KDUP(1),KTR(1),NLV(1),NSLR(10
,1),
     2
ARAS(2,1),ATSA(2,1),RSCRN(2,1),RSINT(2,1),SCFQ(5,1),VOLS(2,1)
,
     3 IINT(1),ICRN(3,1),LJAV(1),LJSV(1),LSAT(1)
      EQUIVALENCE
(DAS,JTR,AWT,BBV,PRB,RTP,SFB,IPNT,JASW,KAW,KDUP,KTR,
     1
NLV,NSLR,ARAS,ATSA,RSCRN,RSINT,SCFQ,VOLS,IINT,ICRN,LJAV,LJSV,
     2 LSAT)
C
C         DYNAMICALLY ALLOCATED COMMON FOR THE PLOT OVERLAY.
      DIMENSION
AMX(4,4,1),COE(6,2,1),CRS(1),JST(2,1),KCL(102,1),KFM(1),
     1 LCL(1),LSG(1),NCS(1),PLB(1),QMX(3,3,2,1),ZST(1)
      EQUIVALENCE
(DAS,AMX,COE,CRS,JST,KCL,KFM,LCL,LSG,NCS,PLB,QMX,ZST)
C
C         DYNAMICALLY ALLOCATED COMMON FOR THE MCPLOT OVERLAY.
      DIMENSION
AB1(1),AB2(1),ERB(1),MCC(1),ORD(1),XCC(1),YCC(1)
      REAL XRR(1),YRR(1)
      EQUIVALENCE (DAS,AB1,AB2,ERB,MCC,ORD,XCC,XRR,YCC,YRR)
C
C----------------------------------------------------------------
C
```

```
      SUBROUTINE IMCN
C           MAIN CODE OF OVERLAY IMCN.
C           INITIATION CODE FOR MONTE CARLO TRANSPORT.
        include 'CM.inc'
        include 'JC.inc'
        CHARACTER BLNK*1
C
C           OPEN SCRATCH FILES FOR COLUMN INPUT.
        HOVR='IMCN'
        OPEN(IU1,STATUS='SCRATCH')
        OPEN(IU2,STATUS='SCRATCH')
C
        IF(KONRUN.NE.0)GO TO 190
C
***********************  INITIAL RUN
************************
*
        C         READ THE REST OF THE INP FILE AND SET UP
DYNAMICALLY
C           ALLOCATED STORAGE.
        DO 5 I=1,MINK*MNK
      5 INK(I)=1
        CALL PASS1
        HOVR='IMCN'
        IF(MODE.EQ.0)KPT(1)=1
        IF(NSR.EQ.6.OR.ISSW.NE.0)CALL SFILES
        IF(NJSR.EQ.0)NJSR=NJSW
        IF(NJSX.EQ.0)NJSX=NJSR
        NDUP(3)=NDUP(3)+NCPARF
        IF(NSR.EQ.71.AND.NSRC.EQ.0)CALL KSRCTP(1)

IF(NSR.EQ.71)MSRK=MAX(4500,NSRCK+NSRCK/2,NSRC/3,MRL,MSRK)
        MLJA=MLJA+2*MXIT*NCOMP
        MLAJ=12*MXA+50
        IF(MODE.NE.2)MXT=MAX(MXT,1)
        IF(MODE.NE.1)NGWW(2-MODE/2)=0
        CALL SETDAS
        IF(LFLL.LT.LICC+4)CALL CHGMEM(LFLL,LICC+4,'IMCN A   ')
C
C           INITIALIZE GENERAL COMMON NOT YET DONE.
        DO 30 I=1,MIPT
        DO 20 J=1,MXDT
     20 DDG(I,J)=HUGE
        DO 30 K=1,2
        DO 30 J=1,MXDX
     30 DDX(I,K,J)=HUGE
        DBCN(10)=DFTINT
        DMP=-240.
        ECF(2)=.001
        EMCF(2)=100.
        EMX=HUGE
        IKZ=5
      DO 35 I=1,MAXF
   35 IVDD(I)=I
      KCY=1
      KTLS=1
      LOST(1)=10
      LOST(2)=10
      NPD=1000
      NTC=50
      NTC1=50
      RKK=1.
      TCO(1)=.001*HUGE
      TCO(2)=.001*HUGE
      WC1(1)=-.5
      WC2(1)=-.25
      IF(MODE.NE.0)WC1(2)=HUGE
      IF(MODE.NE.0)WC2(2)=HUGE
      WGTS(1)=HUGE
      WWP(1,1)=5.
      WWP(2,1)=5.
      WWP(1,3)=5.
      WWP(2,3)=5.
C
C        INITIALIZE DYNAMICALLY ALLOCATED COMMON.
      DO 40 I=1,(LICC+NDP2-1)/NDP2
   40 AAAFD(I)=ZERO
      DO 41 I=LFCDG*NDP2+1,LFCDJ
   41 IIIFD(I)=0
      DO 43 I=LVCDG*NDP2+1,LVCDJ
   43 IIIVD(I)=0
      DO 45 I=LIFL+1,LAWT*NDP2
   45 IFL(I)=0
      DO 47 I=LIPN+1,LICC
   47 JASW(I)=0
      DO 50 I=1,MXA
      IF(NDX(1).NE.0)DXCP(LDXP+1,I)=1.
      IF(NDX(2).NE.0)DXCP(LDXP+2,I)=1.
   50 IF(MODE.EQ.1)GWT(LGWT+I)=-1.
      IF(NGWW(1).NE.0)EWWG(LEWG(1)+NGWW(1))=100.
      IF(NGWW(2).NE.0)EWWG(LEWG(2)+NGWW(2))=100.
      LSC(LLSC+1)=LSCF
      DO 60 I=0,NTAL
   60 IPNT(LIPN+1,1,.)=2+4+64+128
      TRF(LTRF+5,0)=1.
      TRF(LTRF+9,0)=1.
      TRF(LTRF+13,0)=1.
      DO 65 J=1,M)TR
      DO 65 I=5,13
   65 TRF(LTRF+I,J)=HUGE
      DO 70 J=1,MXT
      DO 70 I=1,MXA
   70 TMP(LTMP+J+(I-1)*MXT)=253E-10
C
```

```
C         SORT THE TALLIES, NEUTRONS FIRST.
      DO 100 I=1,NTAL
      JFT(LJFT+I)=1
      K=1
      DO 90 J=1,NTAL
   90 IF(ABS(NTL(J)).LT.ABS(NTL(K)))K=J
      JPTAL(LJPT+1,I)=MOD(NTL(K),1000)
  100 NTL(K)=5000
C
C         PRINT MESSAGE ON VARIABLE DENSITY CODE VERSION
      BLNK=' '
      WRITE(JTTY,*)BLNK
      WRITE(JTTY,'(18x,A42)')MSG1
      WRITE(JTTY,*)BLNK
      WRITE(IUO,*)BLNK
      WRITE(IUO,'(18x,A42)')MSG1
      WRITE(IUO,*)BLNK

C         REREAD AND PRINT THE INP MESSAGE BLOCK AND TITLE
LINE.
      REWIND IUI
      DO 120 IP=1,3
  110 READ(IUI,'(A80)')AID
      ILN=ILN+1
      WRITE(IUO,'(I5,1H-,7X,A80)')ILN,AID
      IF(IP.EQ.2.AND.AID.NE.' ')GO TO 110
  120 IF(IP.EQ.1.AND.AID(1:8).NE.'MESSAGE:')GO TO 130
  130 IF(AID(1:5).EQ.' '.OR.AID(6:72).EQ.' ')GO TO 140
      CALL NXTSYM(AID,' ',6,IT,IU)
      IF(KDATA(AID(1:5)).EQ.2.AND.KDATA(AID(IT:IU)).EQ.2)
     1 CALL ERPRNT(1,2,0,0,0,0,0,0,
     2 '51HTHE TITLE CARD LOOKS SUSPICIOUSLY LIKE A CELL
CARD.')
C
C         REREAD THE REST OF THE INP FILE AND SET UP THE
PROBLEM.
  140 CALL RDPROB

IF(MODE.EQ.1.AND.IFIP(2).EQ.0.AND.NWW(2).EQ.0)WRITE(IUO,150)
  150 FORMAT(/45H PHOTON IMPORTANCES HAVE BEEN SET EQUAL TO
1.)
      CALL IGEOM
      CALL ISOURC
      CALL ITALLY
      CALL VOLUME
      HOVR='IMCN'
      IF(NSR.EQ.40)CALL WTCALC
C
C         WARN OF POSSIBLE NEED FOR SUBROUTINE SRCDX.
      DO 160 I=1,NTAL
  160
IF(JPTAL(LJPT+2,I).EQ.5.AND.JPTAL(LJPT+3,I).EQ.MAX(MODE,1))
```

```
     1 GO TO 170
      IF(NDX(MAX(MODE,1)).EQ.0)GO TO 180
  170 IF(NSR.EQ.0)CALL ERPRNT(1,2,0,0,0,0,0,0,
     1 '58HSUBROUTINE SRCDX IS REQUIRED IF THE SOURCE IS
ANISOTROPIC.')
C
C         WARN OF STRANGE PHOTON TIME CUTOFF.
  180 IF(MODE.EQ.1.AND.TCO(2).NE.TCO(1))CALL
ERPRNT(1,2,0,0,0,0,0,0,
     1 '55HPHOTON TIME CUTOFF IS NOT EQUAL TO NEUTRON TIME
CUTOFF.')
C
C         SET UP THE WEIGHT CUTOFFS.
      IF(WC1(2).EQ.HUGE.AND.MODE.EQ.1)WC1(2)=WC1(1)
      IF(WC1(2).EQ.HUGE.AND.MODE.EQ.2)WC1(2)=-.5
      IF(WC2(2).EQ.HUGE)WC2(2)=.5*WC1(2)
      WCS1(1)=MAX(WC1(1),-WC1(1)*SWTM)
      WCS1(2)=MAX(WC1(2),-WC1(2)*SWTM)
      WCS2(1)=MAX(WC2(1),-WC2(1)*SWTM)
      WCS2(2)=MAX(WC2(2),-WC2(2)*SWTM)
      IF(NDE.NE.0)DBCN(2)=NDE
      CALL UFILES
      CLOSE(IU1)
      CLOSE(IU2)
      RETURN
C
***********************  CONTINUE RUN
***********************
*
        190 CALL TPEFIL(5)
      WRITE(IUO,'(1X,A80)')AID
      IF(NSR.EQ.6.OR.ISSW.NE.0)CALL SFILES
      DO 195 I=1,MINK*MNK
  195 INK(I)=1
      IF(NSR.EQ.71)CALL KSRCTP(3)
      IF(AID1(1:8).NE.'CONTINUE')GO TO 240
C
C         REREAD AND PRINT THE INP MESSAGE BLOCK AND TITLE
LINE.
      REWIND IUI
      WRITE(IUO,'(1H )')
      DO 210 IP=1,3
  200 READ(IUI,'(A80)')KLIN
      ILN=ILN+1
      WRITE(IUO,'(I5,1H-,7X,A80)')ILN,KLIN
      IF(IP.EQ.2.AND.KLIN.NE.' ')GO TO 200
  210 IF(IP.EQ.1.AND.KLIN(1:8).NE.'MESSAGE:')GO TO 220
C
C         READ THE CONTINUE-RUN DATA FROM THE INP FILE.
  220 CALL RDPROB
      HOVR='IMCN'
  240 IF(NDE.NE.0)DBCN(2)=NDE
```

```
       CALL UFILES
       CLOSE(IU1)
       CLOSE(IU2)
       IF((NFER.EQ.0.OR.LFATL.NE.0).AND.JOVR(4).NE.0)CALL
TPEFIL(6)
       RETURN
       END


    SUBROUTINE NEXTIT
C          PROCESS THE NEXT INPUT ITEM.
       include 'CM.inc'
                include 'JC.inc'
                         CHARACTER HT*75
C
C          CHECK THE ITEM BEFORE STORING IT.
       NWC=NWC+1
       CALL CHEKIT
       IF(ICS.LT.0)RETURN
       KS=INDEX('():#',HITM(1:1))
C
       GO TO( 20, 60, 90, 95,
10,100,110,120,130,140,150,161,165,170,280,
     1
180,190,210,216,216,210,218,220,280,280,280,280,280,230,280,
     2
280,280,250,280,280,280,280,280,280,280,280,280,280,280,270,
     3      280,330,340,360,370,380,390,405,410,430)ICA
       GO
TO(440,450,480,490,500,510,520,530,540,550,560,570,580,590,60
0,
     1      610,620,630,670,730,740,810,820, 10,930,
     2      1910,1920,1930)ICA-55
    10 RETURN
C
C >>>>>  CELL DESCRIPTIONS
C          M2C=PREVIOUS SPECIAL CHARACTER:  0=NONE  1=(  2=)
3=:  4=#
C          M3C=FLAG FOR CELL PARAMETERS.
   20 IF(HITM.EQ.'LIKE'.OR.LIKEF.NE.0)GO TO 55
       IF(KS.EQ.0.AND.KITM.EQ.0)M3C=1
       IF(M3C.NE.0)RETURN
C
C          STORE THE MATERIAL NUMBER AND CELL DENSITY.
       IF(NWC.EQ.1)MAT(LMAT+MXA)=IITM
       IF(NWC.EQ.2.AND.MAT(LMAT+MXA).NE.0)RHO(LRHO+MXA)=RITM
C
*=============================================================
*
C          SET ALL CELL DENSITIES TO SEA-LEVEL VALUE [g/cm³]
       IF(NWC.EQ.2.AND.MAT(LMAT+MXA).NE.0) THEN
          IF(RITM.NE.0.) THEN
```

```
             RHO(LRHO+MXA)=-1.225E-3
          ELSE
             RHO(LRHO+MXA)=RITM
          ENDIF
        ENDIF
*
*=====================================================================

      IF(NWC.EQ.1.OR.NWC.EQ.2.AND.MAT(LMAT+MXA).NE.0)GO TO 50
C
C         PREPARE TO STORE LOGICAL OPERATOR OR SURFACE NAME.
      NLJA=NLJA+1
      IF(KS.EQ.0)GO TO 30
C
C       STORE LOGICAL OPERATOR AS 1000000+KS.  KS:  1=(   2=)
3=:  4=#
      LCA(LLCA+MXA)=-ABS(LCA(LLCA+MXA))
      LJA(LLJA+NLJA)=1000000+KS
      GO TO 50

C         STORE THE NAME OF A SURFACE.
   30 LJA(LLJA+NLJA)=IITM
C
   50 M2C=KS
      RETURN
   55 CALL LIKEBT(1)
      IF(NWC.NE.1)RETURN
      DO 57 I=1,NDUP(1)
   57 IF(KDUP(LDUP+I).EQ.ICN+100000)KDUP(LDUP+I)=0
      RETURN
C
C >>>>>  SURFACE DESCRIPTIONS
C         M1C=SURFACE TYPE INDEX.
C         M2C=1 IF SURFACE TYPE SYMBOL IS THE SECOND ITEM.
   60 IF(KITM.NE.0)GO TO 80
      M2C=NWC-1
      DO 70 M1C=1,29
   70 IF(KSF(M1C).EQ.HITM)RETURN
   80 IF(NWC.EQ.1)JTR(LJTR+MXJ)=IITM
      IF(NWC.GT.1)SCF(LSC(LLSC+MXJ)+NWC-M2C-1)=RITM
      RETURN
C
C >>>>>  SPECIFICATION OF COORDINATE TRANSFORMATIONS FOR
SURFACES  TR
   90 TRF(LTRF+1+NWC,MXTR)=RITM
      IF(NWC.LT.4.OR.NWC.GT.12)RETURN
      IF(ICX.EQ.-1)TRF(LTRF+1+NWC,MXTR)=COS(RITM*PIE/180.)

IF(ABS(TRF(LTRF+1+NWC,MXTR)-ANINT(TRF(LTRF+1+NWC,MXTR))).GT.1
E-10)
     1 TRF(LTRF+1,MXTR)=ICN
      RETURN
```

```
C
C >>>>>  VECTORS
  VECT
   95 N=(NWC+3)/4
      IF(NWC.EQ.4*N-3)READ(HITM(2:10),'(BN,I9)')JVC(LJVC+N)
      IF(NWC.NE.4*N-3)VEC(LVEC+NWC-4*N+3,N)=RITM
      RETURN
C
C >>>>>  CELL IMPORTANCES
   IMP
  100 FIM(LFIM+NQW,NWC)=RITM
      IF(IFIP(2).NE.0)RETURN
      IF(MODE.EQ.1.AND.RITM.GT.0.)FIM(LFIM+2,NWC)=1.
      IF(MODE.EQ.2)FIM(LFIM+2,NWC)=FIM(LFIM+1,NWC)
      RETURN
C
C >>>>>  CELL VOLUMES FOR TALLIES
   VOL
  110 IF(KITM.NE.0)VOL(LVOL+NWC)=RITM
      IF(KITM.EQ.0)NOVOL=1
      IF(KITM.EQ.0)NWC=NWC-1
      RETURN
C
C >>>>>  SURFACE AREAS FOR TALLIES
  AREA
  120 ARA(LARA+NWC)=RITM
      RETURN


C >>>>>  PHOTON WEIGHT LOWER BOUNDS
   PWT
  130 GWT(LGWT+NWC)=RITM
      RETURN
C
C >>>>>  EXPONENTIAL TRANSFORM
   EXT
  140 I=1
      IF(HITM(1:1).EQ.'+'.OR.HITM(1:1).EQ.'-')I=2
      IF(HITM(I:I).NE.'S')GO TO 143
      J=I+1
      A=0.
      GO TO 149
  143 DO 145 J=I,NITM
  145 IF(INDEX('VXYZ',HITM(J:J)).NE.0)GO TO 147
  147 HT=HITM(I:J-1)
      READ(HT,'(BN,E21.0)')A
      IF(A.EQ.0.)RETURN
  149 IF(J.EQ.NITM+1)M=4
      IF(J.LT.NITM)READ(HITM(J+1:J+9),'(BN,I9)')M
      IF(J.LT.NITM)M=M+4
      IF(J.EQ.NITM)M=INDEX('XYZ',HITM(J:J))
      QAX(LQAX+NQW,NWC)=M+A
      IF(HITM(1:1).EQ.'-')QAX(LQAX+NQW,NWC)=-M-A
      RETURN
C
C >>>>>  FORCED COLLISIONS
   FCL
  150 FOR(LFOR+NQW,NWC)=RITM
      RETURN
C
C >>>>>  WEIGHT-WINDOW LOWER BOUNDS
   WWN
  161 WWF(LWWF(NQW)+(MAX(1,ICN)-1)*MXA+NWC)=RITM
      RETURN
C
C >>>>>  WEIGHT-WINDOW ENERGIES
   WWE
  165 WWE(LWWE(NQW)+NWC)=RITM
      RETURN
C
C >>>>>  WEIGHT-WINDOW GAME PARAMETERS
   WWP
  170 WWP(NQW,NWC)=RITM
      RETURN
C
C >>>>>  DXTRAN CELL PROBABILITIES
   DXC
  180 DXCP(LDXP+NQW,NWC)=RITM
      RETURN
C
C >>>>>  CELLS WHERE FISSION IS TREATED LIKE CAPTURE
  NONU
  190 LFCL(LLFC+NWC)=IITM-1
      RETURN
C
C >>>>>  SOURCE DISTRIBUTIONS
 SI,DS
C        M1C=DISTRIBUTION INDEX.
C        M2C=CURRENT LOCATION IN SPF.
C        M3C=LOCATION OF N IN KCP.
C        M4C=NWC OF LAST COLON.
  210 IF(KSD(LKSD+20,M1C).EQ.0.OR.M2C.LT.KSD(LKSD+20,M1C))GO
TO 213
      DO 212 IZ=M1C+1,MSD
      I=MSD+M1C+1-IZ
      KSD(LKSD+13,I)=KSD(LKSD+13,I)+4
      M=MAX(KSD(LKSD+4,I),KSD(LKSD+20,I))
      DO 212 J=1,M
      DO 212 K=1,4
  212 SPF(KSD(LKSD+13,I)+K,M+1-J)=SPF(KSD(LKSD+13,I)+K,M-J)
      MXXS=MXXS+4
  213
IF(HITM(1:1).NE.'D'.OR.M4C.NE.0.AND.M4C.EQ.NWC-1.OR.NITM.LT.2
)
```

```
     1 GO TO 214
      IF(KDATA(HITM(2:NITM)).EQ.0)GO TO 214
      IF(KSD(LKSD+13,M1C).EQ.0)KSD(LKSD+13,M1C)=MXXS
      M2C=M2C+1
      READ(HITM(2:4),'(BN,E3.0)')SPF(KSD(LKSD+13,M1C)+1,M2C)
      IF(KSD(LKSD+6,M1C).EQ.0)SPF(KSD(LKSD+13,M1C)+1,M2C)=
     1 SPF(KSD(LKSD+13,M1C)+1,M2C)+100000
      RETURN
  214 IF(KITM.NE.0.OR.HITM(1:1).EQ.'D'.OR.HITM.EQ.':'.OR.
     1 HITM.EQ.'('.OR.HITM.EQ.')')GO TO 215
      NWC=NWC-1
      IF(INDEX('LSFQT',HITM(1:1)).NE.0)KSD(LKSD+5,M1C)=1
      IF(INDEX('SQ',HITM(1:1)).NE.0)KSD(LKSD+6,M1C)=1
      IF(HITM.EQ.'Q')KSD(LKSD+8,M1C)=1
      IF(HITM.EQ.'T')KSD(LKSD+9,M1C)=1
      IF(HITM.EQ.'F')KSD(LKSD+11,M1C)=1
      IF(HITM.EQ.'A')KSD(LKSD+19,M1C)=1
      RETURN
  215 IF(KSD(LKSD+13,M1C).EQ.0)KSD(LKSD+13,M1C)=MXXS
      IF((M4C.EQ.0.OR.M4C.LT.NWC-1).AND.HITM.NE.':'.AND.
     1 HITM.NE.'('.AND.HITM.NE.')')GO TO 2157

IF(HITM.NE.':'.AND.HITM.NE.'('.OR.M4C.EQ.NWC-2.AND.M4C.NE.0)
     1 GO TO 2155
      M3C=MKCP+2
      KCP(LKCP+M3C-1)=-1
      KCP(LKCP+M3C)=1
      MKCP=MKCP+3
      KCP(LKCP+MKCP)=SPF(KSD(LKSD+13,M1C)+1,M2C)
      SPF(KSD(LKSD+13,M1C)+1,M2C)=-LKCP-M3C-1
 2155 IF(HITM.EQ.':')M4C=NWC
      IF(HITM.EQ.':')RETURN
      IF(HITM.EQ.'(')M4C=NWC+3
      KCP(LKCP+M3C)=KCP(LKCP+M3C)+1
      MKCP=MKCP+1
      KCP(LKCP+MKCP)=IITM
      IF(HITM.EQ.'(')KCP(LKCP+MKCP)=1000001
      IF(HITM.EQ.')')KCP(LKCP+MKCP)=1000002
      IF(HITM(1:1).NE.'D')RETURN
      READ(HITM(2:4),'(BN,I3)')KCP(LKCP+MKCP)
      KCP(LKCP+MKCP)=KCP(LKCP+MKCP)+100000
      RETURN
 2157 M2C=M2C+1
      SPF(KSD(LKSD+13,M1C)+1,M2C)=RITM
      RETURN

C >>>>>  SOURCE DISTRIBUTIONS
 SP,SB
C       M1C=DISTRIBUTION INDEX.
C       M2C=FLAG FOR C.
C       M3C=FLAG FOR FUNCTION.
  216 I=INDEX('PB',ICH(2:2))+1
```

```
      IF(KITM.NE.0)GO TO 217
      NWC=NWC-1
      IF(HITM.EQ.'C')M2C=1

IF(HITM.EQ.'C'.AND.KSD(LKSD+19,M1C).EQ.0)KSD(LKSD+19,M1C)=-1
      IF(HITM.EQ.'V')KSD(LKSD+10,M1C)=1
      RETURN
  217 IF(NWC.EQ.1.AND.IITM.LT.0)KSD(LKSD+2,M1C)=IITM
      IF(NWC.EQ.1.AND.IITM.LT.0)M3C=1
      IF(M3C.NE.0)SQQ(LSQQ+NWC+3*I-6,M1C)=RITM
      IF(NWC.EQ.1.AND.IITM.EQ.-21)SQQ(LSQQ+3*I-4,M1C)=12345.
      IF(M3C.NE.0)RETURN
      IF(KSD(LKSD+13,M1C).EQ.0)KSD(LKSD+13,M1C)=MXXS
      SPF(KSD(LKSD+13,M1C)+I,NWC)=RITM
      RETURN
C
C >>>>>  SOURCE COMMENT
    SC
C         M1C=DISTRIBUTION INDEX.
  218 HT=KLIN(6:80)
      DO 219 I=1,75,3
  219
JSCN(MSSC+(I+2)/3)=ICHAR(HT(I:I))*65536+ICHAR(HT(I+1:I+1))*25
6+
     1 ICHAR(HT(I+2:I+2))
      KSD(LKSD+3,M1C)=KSD(LKSD+3,M1C)+25
      MSSC=MSSC+25
      RETURN
C
C >>>>>  SOURCE DEFINITION
  SDEF
C         M1C=NWC OF VARIABLE NAME
C         M2C=INDEX OF CURRENT VARIABLE
C         M3C=INDEX OF DEPENDED-ON VARIABLE OR LOCATION OF N
IN KCP.
C         M4C=NWC OF LAST COLON.
  220 IF(HITM.EQ.'=')NWC=NWC-1
      IF(HITM.EQ.'=')RETURN

IF((HITM.EQ.':'.OR.HITM.EQ.'(').AND.M1C.EQ.NWC)M1C=M1C-2

IF(M4C.NE.0.AND.M4C.EQ.NWC-2.AND.HITM.NE.':'.AND.HITM.NE.'(')
     1 M1C=NWC
      GO TO(221,223,226,227)MIN(4,NWC-M1C+1)
  221 DO 222 M2C=1,MAXV
  222 IF(HITM.EQ.HBLN(M2C,1))RETURN
  223 IF(KITM.NE.0)GO TO 227
      DO 224 M3C=1,MAXF
  224 IF(HITM.EQ.'F'//HBLN(M3C,1))GO TO 225
      GO TO 229
  225 IVDD(M2C)=M3C
      RETURN
  226 IF(M3C.NE.0.AND.HITM.NE.':'.AND.HITM.NE.'(')GO TO 229
  227 IF(M4C.LT.NWC-1.AND.HITM.NE.':'.AND.HITM.NE.'(')GO TO
228
      IF(HITM.EQ.':')M4C=NWC
      IF(NWC.NE.M1C+2)GO TO 2275
      M3C=MKCP+2
      KCP(LKCP+M3C-1)=-1
      KCP(LKCP+M3C)=1
      MKCP=MKCP+3
      KCP(LKCP+MKCP)=SRV(1,M2C)
      SRV(1,M2C)=-LKCP-M3C-1
      IF(IVDIS(1).NE.0)KCP(LKCP+MKCP)=IVDIS(1)+100000
      IVDIS(1)=0
 2275 IF(HITM.EQ.':')RETURN
      IF(HITM.EQ.'(')M4C=NWC+3
      KCP(LKCP+M3C)=KCP(LKCP+M3C)+1
      MKCP=MKCP+1
      KCP(LKCP+MKCP)=IITM
      IF(HITM.EQ.'(')KCP(LKCP+MKCP)=1000001
      IF(HITM.EQ.')')KCP(LKCP+MKCP)=1000002
      IF(HITM(1:1).NE.'D')RETURN
      READ(HITM(2:4),'(BN,I3)')KCP(LKCP+MKCP)
      KCP(LKCP+MKCP)=KCP(LKCP+MKCP)+100000
      RETURN
  228 SRV(NWC-M1C,M2C)=RITM
      IF(NWC-M1C.LT.NVS(M2C))RETURN
  229 M1C=NWC+1
      M3C=0
      RETURN
C
C >>>>>  TALLY COMMENT
    FC
  230 HT=KLIN(6:80)
      DO 240 I=1,75,3
  240 RTP(LRTP+IPL+(NWC-1)*25+(I+2)/3)=ICHAR(HT(I:I))*65536+
     1 ICHAR(HT(I+1:I+1))*256+ICHAR(HT(I+2:I+2))
      RETURN
C
C >>>>>  ORDER OF TALLY PRINTING
    FQ
  250 K=INDEX('FDUSMCET',HITM(1:1))
      DO 260 I=1,7

IF(JFQ(LJFQ+I,ITAL).EQ.K)JFQ(LJFQ+I,ITAL)=JFQ(LJFQ+I+1,ITAL)
  260
IF(JFQ(LJFQ+I+1,ITAL).EQ.JFQ(LJFQ+I,ITAL))JFQ(LJFQ+I+1,ITAL)=
K
      JFQ(LJFQ+8,ITAL)=-K
      RETURN
C
C >>>>>  TALLY FLUCTUATION CHART BINS
    TF
  270 JTF(LJTF+NWC,ITAL)=IITM
      RETURN
C
C >>>>>  OTHER TALLY CARDS
PD,F,FX,FY,FZ,FT,E,T,C,FM,DE,
C
DF,EM,TM,CM,CF,SF,FS,SD,FU,DD
C          M2C=NWC OF LAST FT-CARD PARAMETER COUNT.
C          SPECIAL HANDLING FOR DXTRAN DD CARD.
  280 IF(ICH.NE.'DD'.OR.ICN.GT.2)GO TO 300
      I=2-MOD(NWC,2)
      J=(NWC+1)/2
      IF(ICN.EQ.0)GO TO 290
      DDX(ICN,I,J)=RITM
      RETURN
  290 IF(DDX(1,I,J).EQ.HUGE)DDX(1,I,J)=RITM
      IF(DDX(2,I,J).EQ.HUGE)DDX(2,I,J)=RITM
C
  300 IF(KITM.EQ.0)GO TO 310
      RTP(LRTP+IPL+NWC)=RITM
      RETURN
  310 IF(ICH.EQ.'FT')GO TO 323
      N=INDEX('NT',HITM(1:1))
      IF(N.EQ.0)GO TO 320
      I=(INDEX('FU FS FM C  E  T  ',ICH(1:3))+5)/3
      IF(I.EQ.1.AND.MOD(ICN,10).NE.5)I=0
      K=2**I
      L=MOD(IPNT(LIPN+1,1,ITAL)/K,2)

IF(L.NE.N-1)IPNT(LIPN+1,1,ITAL)=IPNT(LIPN+1,1,ITAL)+(2*N-3)*K
      NWC=NWC-1
      RETURN
  320 IF(KS.NE.0)RTP(LRTP+IPL+NWC)=1000000+KS
      IF(ICH.NE.'DE'.AND.ICH.NE.'DF')RETURN
      IF(HITM.EQ.'LIN')RTP(LRTP+IPL+NWC)=-1.
      IF(HITM.EQ.'LOG')NWC=NWC-1
      RETURN
  323 DO 325 I=1,MKFT
  325 IF(HITM.EQ.HFT(I))RTP(LRTP+IPL+NWC)=I
      IF(M2C.NE.0)RTP(LRTP+IPL+M2C)=NWC-M2C-1
      NWC=NWC+1
      M2C=NWC
      RETURN
C
C >>>>>  DXTRAN PARAMETERS
   DXT
  330 J=MOD(NWC-1,5)+1
      K=(NWC+4)/5
      IF(K.LE.MXDX)DXX(NQW,J,K)=RITM
      IF(K.LE.MXDX.AND.J.GE.4)DXX(NQW,J,K)=(RITM*1.00001)**2
      IF(J.LE.3)DXW(NQW,J)=RITM
      IF(J.NE.4)RETURN
      DXW(NQW,1)=0.
      DXW(NQW,2)=0.
      DXW(NQW,3)=0.
      RETURN
C
C >>>>>  MATERIAL SPECIFICATIONS
   M
  340 IF(MOD(NWC,2).EQ.0)GO TO 350
      MIX=MIX+1
      HT=HITM
      IF(INDEX(HT,'.').EQ.0)HT(NITM+1:NITM+1)='.'
      HMM(MIX)(8-INDEX(HT,'.'):10)=HT
      IF(HMM(MIX)(8:8).NE.' '.AND.HMM(MIX)(9:9).EQ.'
')HMM(MIX)(9:9)='0'
      IF(HMM(MIX)(8:10).EQ.'0  '.OR.HMM(MIX)(8:10).EQ.'00
'.OR.
     1 HMM(MIX)(8:10).EQ.'000')HMM(MIX)(8:10)='   '
      RETURN
  350 FME(MIX)=RITM
      IF(RITM.EQ.0.)HMM(MIX)=' '
      IF(RITM.EQ.0.)MIX=MIX-1
      RETURN
C >>>>>  NUCLIDES FOR DISCRETE TREATMENT
    DRXS
  360 HT=HITM
      IF(INDEX(HT,'.').EQ.0)HT(NITM+1:NITM+1)='.'
      HDR(NWC)(8-INDEX(HT,'.'):10)=HT
      RETURN
C
C >>>>>  TOTAL OR PROMPT NUBAR
 TOTNU
  370 ITOTNU=2
      RETURN
C
C >>>>>  ATOMIC WEIGHTS
 AWTAB
  380 IF(MOD(NWC,2).EQ.1)KAW(LKAW+(NWC+1)/2)=IITM
      IF(MOD(NWC,2).EQ.0)AWT(LAWT+NWC/2)=RITM
      RETURN
C
C >>>>>  CROSS-SECTION DIRECTORY INFORMATION
   XS
C        M1C=LAST CHARACTER POSITION USED SO FAR IN
XSCRD(NXSC).
  390 IF(NWC.NE.1)GO TO 400
      NXSC=NXSC+1
      XSCRD(NXSC)=' '
      WRITE(XSCRD(NXSC)(MXC-3:MXC),'(I4)')ICN
      XSCRD(NXSC)(8-INDEX(HITM,'.'):10)=HITM
      M1C=10
      RETURN
  400 XSCRD(NXSC)(M1C+2:M1C+NITM+1)=HITM
```

```
      M1C=M1C+NITM+1
      RETURN
C
C >>>>>  VOID CELLS
  VOID
  405 IOID=0
      I=NAMCHG(1,IITM)
      RHO(LRHO+I)=0.
      MAT(LMAT+I)=0
      RETURN
C
C >>>>>  PHYSICS PARAMETERS
  PHYS
  410 IF(NQW.EQ.2)GO TO 420
      IF(NWC.EQ.1)EMX=RITM
      IF(NWC.EQ.2)EMCF(1)=RITM
      RETURN
  420 EMCF(2)=MAX(RITM,ZERO+.001)
      RETURN
C
C >>>>>  ENERGY SPLITTING
 ESPLT
  430 ESPL(NQW,NWC)=RITM
      RETURN
C
C >>>>>  THERMAL TEMPERATURES
   TMP
  440 TMP(LTMP+MAX(1,ICN)+(NWC-1)*MXT)=RITM
      RETURN
C
C >>>>>  THERMAL TIMES
 THTME
  450 TTH(LTTH+NWC)=RITM
      RETURN
C >>>>>  THERMAL S(A,B) DATA SPECIFICATIONS
    MT
  480 INDT=INDT+1
      IMTX(INDT)=ICN
      HT=HITM
      IF(INDEX(HT,'.').EQ.0)HT(NITM+1:NITM+1)='.'
      HMT(INDT)(8-INDEX(HT,'.'):10)=HT
      IF(HMT(INDT)(8:8).NE.' '.AND.HMT(INDT)(9:9).EQ.' ')
     1 HMT(INDT)(9:9)='0'
      RETURN
C
C >>>>>  CUTOFFS
   CUT
  490 IF(NWC.EQ.1.AND.RITM.NE.0.)TCO(NQW)=RITM

IF(NWC.EQ.1.AND.RITM.NE.0..AND.TCO(2).EQ..001*HUGE)TCO(2)=RIT
M
      IF(NWC.EQ.2)ECF(NQW)=MAX(ZERO,RITM)
      IF(NWC.EQ.3)WC1(NQW)=RITM
      IF(NWC.EQ.3)WC2(NQW)=.5*WC1(NQW)
      IF(NWC.EQ.4)WC2(NQW)=SIGN(RITM,WC1(NQW))
      IF(NWC.EQ.5)SWTM=RITM
      IF(NWC.EQ.5.AND.RITM.EQ.0.)SWTM=-1.
      RETURN
C
C >>>>>  SOURCE PARTICLE CUTOFF NUMBER
   NPS
  500 NPP=IITM
      RETURN
C
C >>>>>  COMPUTER TIME CUTOFF
  CTME
  510 CTME=RITM
      RETURN
C
C >>>>>  INTEGER QUANTITIES FOR TEMPORARY CODE FEATURES
  IDUM
  520 IDUM(NWC)=IITM
      RETURN
C
C >>>>>  REAL QUANTITIES FOR TEMPORARY CODE FEATURES
  RDUM
  530 RDUM(NWC)=RITM
      RETURN
C
C >>>>>  PRINT AND DUMP CONTROLS
 PRDMP
  540 IF(NWC.EQ.1)PRN=RITM
      IF(NWC.EQ.2)DMP=RITM
      IF(NWC.EQ.3)MCT=IITM
      RETURN
C
C >>>>>  TERMINATION AND PRINT CONTROL FOR LOST PARTICLES
  LOST   550 LOST(NWC)=IITM
      RETURN
C
C >>>>>  DEBUGGING CONTROLS
  DBCN
  560 DBCN(NWC)=RITM
      RETURN
C
C >>>>>  SPECIFICATIONS FOR USER FILES
 FILES
  570 J=(NWC+4)/5
      N=NWC-5*(J-1)
      IF(N.EQ.1)KUFIL(1,J)=IITM
      IF(N.EQ.2)UFIL(1,J)=HITM
      IF(N.EQ.3)UFIL(2,J)=HSD(INDEX('SD',HITM(1:1)))
      IF(N.EQ.4)UFIL(3,J)=HFU(INDEX('FU',HITM(1:1)))
      IF(N.EQ.5)KUFIL(2,J)=IITM
      RETURN
C
C >>>>>  PRINT CONTROL
 PRINT
  580 IF(MNK.EQ.1)RETURN
      IF(IITM.LT.0)GO TO 581
      INK(IITM)=1
      RETURN
  581 IF(MNK.NE.0)GO TO 585
      MNK=-1
      DO 583 I=1,MINK
  583 INK(I)=1
  585 INK(-IITM)=0
      RETURN
C
C >>>>>  KCODE SPECIFICATIONS
 KCODE
  590 IF(NWC.EQ.4)KCT=RITM
      IF(NWC.EQ.6.AND.RITM.NE.0.)KNRM=1
      IF(NWC.EQ.2.AND.KONRUN.EQ.0.AND.RITM.NE.0.)RKK=RITM
      IF(NWC.EQ.3.AND.KONRUN.EQ.0)IKZ=RITM
      RETURN
C
C >>>>>  LOCATIONS OF KCODE SOURCE POINTS
  KSRC
  600 FSO(LFSO+NWC+2*((NWC-1)/3))=RITM
      RETURN
C
C >>>>>  WEIGHT-WINDOW GENERATOR PARAMETERS
   WWG
  610 WWG(NWC)=RITM
      RETURN
C
C >>>>>  ENERGY BINS FOR WEIGHT-WINDOW GENERATOR
  WWGE
  620 EWWG(LEWG(NQW)+NWC)=RITM
      RETURN
C
C >>>>>  SURFACE SOURCE WRITE INFORMATION
   SSW
C         M1C=NWC OF VARIABLE NAME.
C         M2C=INDEX OF CURRENT VARIABLE.
  630 IF(NWC.GT.NJSS)GO TO 640
      M1C=NJSS+1
      JSS(LJSS+NWC)=IITM
      RETURN
C
C         SEARCH FOR KEYWORDS AFTER ALL SURFACE NUMBERS ARE
READ.
  640 IF(HITM.EQ.'=')NWC=NWC-1
      IF(HITM.EQ.'=')RETURN
      IF(KITM.NE.0)GO TO 660
      M1C=NWC
      DO 650 M2C=1,MAXW
  650 IF(HITM.EQ.HBLW(M2C))RETURN
  660 IF(M2C.EQ.1)NSPH=IITM
      IF(M2C.EQ.2)IPTY=IITM
      RETURN
C
C >>>>>  SURFACE SOURCE READ INFORMATION
   SSR
C          M1C=NWC OF VARIABLE NAME.
C          M2C=INDEX OF CURRENT VARIABLE.
  670 IF(HITM.EQ.'=')NWC=NWC-1
      IF(HITM.EQ.'=')RETURN
      IF(NWC.GT.M1C)GO TO 690
      DO 680 M2C=1,MAXV
  680 IF(HITM.EQ.HBLN(M2C,2))RETURN
  690 IF(KITM.EQ.0)GO TO 720
      IF(M2C.GT.3)GO TO 710
      IF(M2C.GT.2)GO TO 700
      JASR(LJAR+NWC-M1C)=IITM
      IF(NWC-M1C.LT.NJSR)RETURN
      GO TO 720
  700 ISS(LISS+NWC-M1C)=IITM
      IF(NWC-M1C.LT.NJSX)RETURN
      GO TO 720
  710 SRV(NWC-M1C,M2C)=RITM
      IF(NWC-M1C.LT.NVS(M2C))RETURN
  720 M1C=NWC+1
      RETURN
C
C >>>>>  UNIVERSE DESIGNATORS
    U
  730 JUN(LJUN+NWC)=IITM
      RETURN
C
C >>>>>  CELL TRANSFORMATIONS
  TRCL
C          M1C=CELL INDEX
C          M2C=NWC OF LAST CELL SEEN, NEGATIVE IF LEFT PARENS.
  740 GO TO(750,770,780)1+INDEX('()',HITM(1:1))
  750 IF(M2C.LT.0)GO TO 760
      M1C=M1C+NWC-M2C
      KTR(LKTR+M1C)=IITM
      M2C=NWC
      RETURN
  760 M=NWC+M2C+1
      TRF(LTRF+M,MXTR)=RITM
      IF(M.LT.5.OR.M.GT.13)RETURN
      IF(ICX.EQ.-1)TRF(LTRF+M,MXTR)=COS(RITM*PIE/180.)

IF(ABS(TRF(LTRF+M,MXTR)-ANINT(TRF(LTRF+M,MXTR))).GT.1E-10)
     1 TRF(LTRF+1,MXTR)=ABS(TRF(LTRF+1,MXTR))
```

```
      RETURN
  770 M1C=M1C+NWC-M2C
      MXTR=MXTR+1
      KTR(LKTR+M1C)=1000+MXTR
      TRF(LTRF+1,MXTR)=-1000-MXTR
      M2C=-NWC
      RETURN
  780 IF(NWC.NE.-M2C+2)GO TO 790
      KTR(LKTR+M1C)=TRF(LTRF+2,MXTR)
      TRF(LTRF+2,MXTR)=HUGE
      MXTR=MXTR-1
      GO TO 800
  790 CALL TRFMAT(MXTR)
  800 M2C=NWC
      RETURN
C
C >>>>>  LATTICE TYPE
   LAT
  810 IF(IITM.EQ.0)RETURN
      LAT(LLAT+1,NWC)=IITM
      NLAT=NLAT+1
      LAT(LLAT+2,NWC)=NLAT
      RETURN
C
C >>>>>  CELL-FILLING UNIVERSES, WITH TRANSFORMATIONS
  FILL
C        M1C=CELL INDEX
C        M2C(POSITIVE)=NWC OF LAST ITEM SEEN, NOT IN PARENS
C        M2C(NEGATIVE)=-NWC OF LEFT PARENS, WHEN IN PARENS
C        M3C=0 WHEN NOT IN LATTICE FILL
C        M3C(NEGATIVE)=-NWC OF I2 IN I1:I2
C        M3C(POSITIVE)=N OF LAF(LLAF+M,N)
C        M4C=MAXIMUM VALUE OF N
  820 GO TO(830,880,890,900)1+INDEX(':()',HITM(1:1))
  830 IF(M2C.LT.0)GO TO 870
      IF(M3C.EQ.0)GO TO 860
      IF(M3C.GT.0)GO TO 840
      I=NWC+M3C
      IF(MOD(I,3).NE.0)LAF(LLAF+MLAF+2+I/3,1)=IITM
      IF(MOD(I,3).EQ.0)LAF(LLAF+MLAF+1+I/3,2)=IITM-
     1 LAF(LLAF+MLAF+1+I/3,1)+1
      IF(I.NE.6)RETURN
      M3C=2
      M2C=NWC
      M4C=2+LAF(LLAF+MLAF+1,2)*LAF(LLAF+MLAF+2,2)*
     1 LAF(LLAF+MLAF+3,2)
      RETURN
  840 M3C=M3C+NWC-M2C
      IF(M3C.GT.M4C)GO TO 850
      LAF(LLAF+MLAF+1,M3C)=IITM
      M2C=NWC
      RETURN
  850 MLAF=MLAF+M4C*3
      M3C=0
  860 M1C=M1C+NWC-M2C
      MFL(LMFL+1,M1C)=IITM
      M2C=NWC
      RETURN
  870 M=NWC+M2C+1
      TRF(LTRF+M,MXTR)=RITM
      IF(M.LT.5.OR.M.GT.13)RETURN
      IF(ICX.EQ.-1)TRF(LTRF+M,MXTR)=COS(RITM*PIE/180.)

IF(ABS(TRF(LTRF+M,MXTR)-ANINT(TRF(LTRF+M,MXTR))).GT.1E-10)
     1 TRF(LTRF+1,MXTR)=ABS(TRF(LTRF+1,MXTR))
      RETURN
  880 IF(M3C.NE.0)RETURN
      M3C=-NWC-1
      LAF(LLAF+MLAF+1,1)=MFL(LMFL+1,M1C)
      MFL(LMFL+1,M1C)=-LLAF-MLAF
      RETURN
  890 MXTR=MXTR+1
      IF(M3C.EQ.0)MFL(LMFL+3,M1C)=1000+MXTR
      IF(M3C.NE.0)LAF(LLAF+MLAF+3,M3C)=1000+MXTR
      TRF(LTRF+1,MXTR)=-1000-MXTR
      M2C=-NWC
      RETURN
  900 IF(NWC.NE.-M2C+2)GO TO 910
      IF(M3C.EQ.0)MFL(LMFL+3,M1C)=TRF(LTRF+2,MXTR)
      IF(M3C.NE.0)LAF(LLAF+MLAF+3,M3C)=TRF(LTRF+2,MXTR)
      TRF(LTRF+2,MXTR)=HUGE
      MXTR=MXTR-1
      GO TO 920
  910 CALL TRFMAT(MXTR)
  920 M2C=NWC
      RETURN
C
C >>>>>  PHOTON-PRODUCTION BIAS
 PIKMT
  930 NPIKMT=NPIKMT+1
      PIK(LPIK+NPIKMT)=RITM
      RETURN
C
C >>>>>  FIRST SPARE CARD TYPE
    ZA
 1910 CONTINUE
      RETURN
C
C >>>>>  SECOND SPARE CARD TYPE
    ZB
 1920 CONTINUE
      RETURN
C
C >>>>>  THIRD SPARE CARD TYPE
```

```
     ZC
  1930 CONTINUE
       RETURN
C
       END


    SUBROUTINE HSTORY
C          RUN THE COMPLETE HISTORY OF A SOURCE PARTICLE.
       include 'CM.inc'
       include 'RC.inc'
C
C          DEBUG FEATURES:  SET UP EVENT LOG.  PRINT DEBUG
LINE.
       KRFLG=0
       IF(NPS+1.GE.DBCN(3).AND.NPS+1.LE.DBCN(4))KRFLG=1
       IF(DBCN(2).EQ.0.)GO TO 20

IF(MOD(NPS+1,INT(DBCN(2))).EQ.0)WRITE(IUO,10)NPS+1,NCT(1)+NCT
(2),
      1 NRN,RIJK
    10 FORMAT(5H DBCN,3I9,4X,F16.0,TL1,1H )
C
C          SAVE VARIABLE COMMON FOR POSSIBLE LOST PARTICLE
RERUN.
    20 DO 25 I=1,NVARCM
    25 GVBU(I)=GVARCM(I)
       DO 30 I=1,LVARCM
    30 JVBU(I)=JVARCM(I)
C
C          START A PARTICLE FROM THE SOURCE.
    40 CALL STARTP
       IF(NTER.NE.0)GO TO 310
       IF(KDB.NE.0)GO TO 410
       IF(NST.NE.0) then
       RETURN
       end if
C
C          TERMINATE THE PARTICLE IF ITS ENERGY IS BELOW
CUTOFF.
C          MOST BANKED PARTICLES COME BACK HERE.
    60 IF(ERG.LT.ECF(IPT).EQV.MCAL.LT.2)GO TO 270
C
C          CALCULATE THE DISTANCE TO THE CELL BOUNDARY, DLS.
       IF(LCA(LLCA+ICL).LT.0)CALL CHKCEL(ICL,3,J)
    70 IF(WGT.LE.0.)CALL EXPIRE(1,'HSTORY',
      1 'THE WEIGHT OF THE CURRENT PARTICLE IS ZERO OR LESS.')
       CALL TRACK(ICL)
       IF(KDB.NE.0)GO TO 410
C
C          CALCULATE THE DISTANCE TO THE NEAREST DXTRAN SPHERE,
DXL.
```

```
      DXL=HUGE
      DO 80 I=1,NDX(IPT)
      IF(IDX.EQ.I)GO TO 80
      F=DXX(IPT,1,I)-XXX
      G=DXX(IPT,2,I)-YYY
      H=DXX(IPT,3,I)-ZZZ
      Q=F*UUU+G*VVV+H*WWW
      C=MIN(MAX(ZERO,Q),DXL)

IF((F-UUU*C)**2+(G-VVV*C)**2+(H-WWW*C)**2.LT.DXX(IPT,5,I))
     1
DXL=MIN(DXL,Q-SQRT(MAX(ZERO,Q**2+DXX(IPT,5,I)-F**2-G**2-H**2)
))
   80 CONTINUE
C

C         CALCULATE THE DISTANCE TO TIME CUTOFF, DTC.
      DTC=VEL*(TCO(IPT)-TME)

C         CALCULATE THE CROSS SECTIONS IN THIS CELL.
      TOTM=0.
      PFP=0.
      STP=0.
      DEB=HUGE
      IF(MLL(LMLL+1,ICL).EQ.0)GO TO 85
      IF(IPT.EQ.1)CALL ACETOT
      IF(IPT.EQ.2)CALL PHOTOT
C
C         SPECIAL TREATMENT FOR MULTIGROUP ELECTRONS.
      IF(MCAL.EQ.0)GO TO 85
      PFP=10.*PFP
      IF(STP.EQ.0.)GO TO 85
      M=JXS(1,MGEGBT(1))+JGP-1
      IF(MCAL.EQ.1)DEB=(ERG-YSS(M)+.5*YSS(M+JGM(1)))/STP
      IF(MCAL.EQ.2)DEB=(YSS(M)+.5*YSS(M+JGM(1))-ERG)/STP
C
C         CALCULATE THE MEAN FREE PATH, GS, AND ITS
RECIPROCAL, QPL.
   85 GS=0.
      PMF=HUGE
      QPL=(TOTM+PFP)*RHO(LRHO+ICL)
      PLE=QPL
      IF(PLE.EQ.0.)GO TO 160
      PFP=PFP/(TOTM+PFP)
      IF(QAX(LQAX+IPT,ICL).NE.0.)CALL EXTRAN
      IF(QPL.LE.0.)GO TO 160
      GS=1./QPL
C
C         DECIDE WHETHER TO FORCE A COLLISION.
      IF(FOR(LFOR+IPT,ICL).EQ.0)GO TO 90
      CALL FORCOL
      IF(NTER.NE.0)GO TO 310
      GO TO 160
C
C          SAMPLE THE DISTANCE TO COLLISION, PMF, NORMALLY.
cr  move st # 90 to line below
   90 CONTINUE
        qzridq=RANG()
        qzridq1=-LOG(qzridq)
      PMF=qzridq1*GS

*==========================================================================
*
*    THIS PORTION OF CODE CORRECTS FOR A VARIABLE DENSITY
*    EXPONENTIAL ATMOSPHERE
*
*==========================================================================
*
C    CALCULATE A NEW DISTANCE TO COLLISION, PMF, BASED ON A
C       VARIABLE DENSITY ATMOSPHERE.
      NINP=0
      CALL EQDIST(PMF, D, ZZZ, WWW, NINP)

C          PARTICLE EXITS TOP OR BOTTOM OF ATMOSPHERE IF PMF=-1
      IF (PMF.EQ.-1)PMF=HUGE
C
C          RE-CALCULATE THE MEAN FREE PATH, GS, AND ITS
C          RECIPROCAL, QPL.
      GS=PMF/qzridq1
      QPL=1./GS
      PLE=QPL
*
*==========================================================================
C
C          TALLY THE TRACK LENGTH IN THE CELL.
  160 D=MIN(PMF,DLS,DXL,DTC,DEB)
      IF(NSR.NE.71)GO TO 180
      IF(IPT.NE.1.OR.LFCL(LLFC+ICL).EQ.0)GO TO 180
      FM=0.
      DO 170 M=MLL(LMLL+1,ICL),MLL(LMLL+2,ICL)
  170 FM=FM+RTCR(10,LME(1,M))*RTCR(8,LME(1,M))*FME(M)
      SUMK(3)=SUMK(3)+FM*D*WGT*RHO(LRHO+ICL)
  180 L=LOCCT(LLCT+IPT,ICL)
      IF(L.NE.0)CALL TALLY(L,D)
      DO 185 I=0,LEV-1
      L=LOCCT(LLCT+IPT,INT(UDT(7,I)))
  185 IF(L.NE.0)CALL TALLY(L,D)
      JSU=JAP
C
C          INCREMENT THE SUMMARY ACCOUNTS.
      DT=D/VEL
      PAC(LPAC+IPT,5,ICL)=PAC(LPAC+IPT,5,ICL)+WGT*DT*ERG
      PAC(LPAC+IPT,6,ICL)=PAC(LPAC+IPT,6,ICL)+WGT*D*ERG
      PAC(LPAC+IPT,7,ICL)=PAC(LPAC+IPT,7,ICL)+D
```

```
IF(PLE.NE.O.)PAC(LPAC+IPT,8,ICL)=PAC(LPAC+IPT,8,ICL)+WGT*D/PL
E
      PAC(LPAC+IPT,9,ICL)=PAC(LPAC+IPT,9,ICL)+WGT*DT
      PAC(LPAC+IPT,10,ICL)=PAC(LPAC+IPT,10,ICL)+WGT*D
C
C         UPDATE THE PARTICLE TO THE SURFACE, COLLISION, OR
TERMINATION.
C         BANKED UNCOLLIDED PART COMES BACK HERE.
  190 TME=TME+DT
      XXX=XXX+UUU*D
      YYY=YYY+VVV*D
      ZZZ=ZZZ+WWW*D
      DO 195 L=0,LEV-1
      UDT(1,L)=UDT(1,L)+UDT(4,L)*D
      UDT(2.L)=UDT(2,L)+UDT(5,L)*D
  195 UDT(3,L)=UDT(3,L)+UDT(6,L)*D
C
C         SPECIAL TREATMENT FOR MULTIGROUP ELECTRONS.
      IF(STP.EQ.O.)GO TO 197
      T1=D*STP
      IF(MCAL.EQ.2)T1=-T1
      ERG=ERG-T1
      PAX(IPT,6,2)=PAX(IPT,6,2)+T1*WGT
      IF(ERG.LE.ECF(1).EQV.MCAL.LT.2)GO TO 270
      RM=YSS(JXS(1,MGEGBT(1))+JGP-1+2*JGM(1))
      VEL=SLITE*SQRT(ERG*(ERG+2.*RM))/(ERG+RM)
  197 EGO=ERG
C
C         SCORE FLUX IN CELL FOR MULTIGROUP WEIGHT-WINDOW
GENERATION.
      IF(ICW.NE.O)FLX(LFLX(IPT)+MXA*(JGP-1)+ICL)=
     1 FLX(LFLX(IPT)+MXA*(JGP-1)+ICL)+D*WGT
      IF(D.EQ.DTC)GO TO 280
      IF(D.EQ.DXL)GO TO 300
C
C         PROCESS DXTRAN PARTICLE AS IT LEAVES ITS SPHERE.
      IF(IDX.EQ.O)GO TO 200
      IF((XXX-DXX(IPT,1,IDX))**2+(YYY-DXX(IPT,2,IDX))**2+
     1 (ZZZ-DXX(IPT,3,IDX))**2.LT.DXX(IPT,5,IDX))GO TO 200
      IDX=O
      IF(WWP(IPT,4).NE.O.)GO TO 200
      IF(WGT*FIM1.GT.FIS*WCS2(IPT))GO TO 200
      T1=WCS1(IPT)*FIS/FIM1
      IF(T1.EQ.O.)GO TO 200
      IF(WGT.LT.T1*RANG())GO TO 290
      PWB(LPWB+IPT,10,ICL)=PWB(LPWB+IPT,10,ICL)+T1-WGT
      PAX(IPT,2,6)=PAX(IPT,2,6)+T1-WGT
      PAX(IPT,3,6)=PAX(IPT,3,6)+(T1-WGT)*ERG
      WGT=T1
C
C         ADJUST THE WEIGHT FOR EXPONENTIAL TRANSFORMATION.
  200 IF(QAX(LQAX+IPT,ICL).EQ.O.)GO TO 210
```

```
      T1=WGT
      WGT=WGT*EXP((QPL-PLE)*D)
      IF(PMF.LT.DLS)WGT=WGT*PLE*GS
      PWB(LPWB+IPT,12,ICL)=PWB(LPWB+IPT,12,ICL)-(T1-WGT)
      I=2
      IF(T1.GT.WGT)I=5
      PAX(IPT,I,11)=PAX(IPT,I,11)+ABS(T1-WGT)
      PAX(IPT,I+1,11)=PAX(IPT,I+1,11)+ABS(T1-WGT)*ERG
C
C        PROCESS THE PARTICLE THRU THE CELL BOUNDARY IF NO
COLLISION.
  210 IF(D.NE.DLS)GO TO 220
      CALL SURFAC
      IF(KRFLG.NE.0)CALL EVENTP(3)
      IF(KDB.NE.0)GO TO 410
      IF(NTER.NE.0)GO TO 310
      GO TO 70
C
C        CALCULATE EVERYTHING ABOUT THE COLLISION.
  220 PAC(LPAC+IPT,3,ICL)=PAC(LPAC+IPT,3,ICL)+1.
      PAC(LPAC+IPT,4,ICL)=PAC(LPAC+IPT,4,ICL)+WGT
      NCH(IPT)=NCH(IPT)+1
      NCP=NCP+1
      IF(NCH(IPT).EQ.DBCN(9))GO TO 260
  230 IF(IPT.EQ.1)CALL COLIDN
      IF(IPT.EQ.2)CALL COLIDP
      IF(KDB.NE.0)GO TO 410
      IF(NTER.NE.0)GO TO 310


C        TALLY DETECTORS AND CREATE DXTRAN PARTICLES.
      IF(NDET(IPT).NE.0)CALL TALLYD
      IF(KDB.NE.0)GO TO 410
      IF(NDX(IPT).EQ.0)GO TO 255
      IF(NDX(IPT).GT.1.OR.IDX.EQ.0)CALL DXTRAN
      IF(KDB.NE.0)GO TO 410
C
C        PLAY THE WEIGHT-WINDOW AND ENERGY-SPLITTING GAMES.
  255 IF(ABS(WWP(IPT,4)).EQ.1..AND.IDX.EQ.0)CALL WTWNDO(WW)
      IF(NTER.NE.0)GO TO 310
      IF(ESPL(IPT,1).NE.0.)CALL ERGIMP
      IF(NTER.NE.0)GO TO 310
      GO TO 60
C
C        DEBUG FEATURE:  COLLISION LOOP BREAKPOINT.
  260 GO TO 230
C
***************  PROCESS TERMINATED PARTICLES.
****************
*
        270 NTER=4
      GO TO 310
  280 NTER=13
      GO TO 310
  290 NTER=6
      GO TO 310
  300 NTER=10
C
C         INCREMENT PARTICLE STATISTICS FOR TERMINATION TYPE
NTER.
  310 IF(KRFLG.NE.0)CALL EVENTP(5)
      IF(IGWW.NE.0.AND.(NTER.LT.6.OR.NTER.GT.9))CALL
WGTWWG(1,WGT)
      J=JRWB(NTER)
      IF(J.NE.0)PWB(LPWB+IPT,J,ICL)=PWB(LPWB+IPT,J,ICL)-WGT
      IF(NTER.EQ.1)TMAV(IPT,1)=TMAV(IPT,1)+TME*WGT
      IF(NTER.EQ.3)TMAV(IPT,2)=TMAV(IPT,2)+TME*WGT
      TMAV(IPT,3)=TMAV(IPT,3)+TME*WGT
      PAX(IPT,4,NTER)=PAX(IPT,4,NTER)+1.
      PAX(IPT,5,NTER)=PAX(IPT,5,NTER)+WGT
      PAX(IPT,6,NTER)=PAX(IPT,6,NTER)+WGT*ERG
      IF(NSR.NE.71)GO TO 320
      IF(IPT.NE.1)GO TO 320
      IF(NTER.NE.1.AND.NTER.NE.4.AND.NTER.NE.13)GO TO 320
      RLT(1)=RLT(1)+WGT*TME
      RLT(2)=RLT(2)+WGT*TME
  320 NTER=0
C
C         GET THE NEXT PARTICLE FROM THE BANK, IF THERE ARE
ANY.
      IF(NBNK.EQ.0)GO TO 390
      DO 330 I=1,MXA
  330 IPAC2(LPC2+I)=0
      CALL BANKIT(2)
 100  IF(KFL.EQ.0)GO TO 370
      IF(KFL.EQ.1)GO TO 350
      DO 340 I=1,MXA
  340 IF(IFL(LIFL+I).GE.NODE)IFL(LIFL+I)=0
      IF(KFL.EQ.2)GO TO 370
  350 DO 360 J=1,MXJ
  360 IF(JFL(LJFL+J).GE.NODE)JFL(LJFL+J)=0
  370 IF(KRFLG.NE.0)CALL EVENTP(2)
      PAC(LPAC+IPT,2,ICL)=PAC(LPAC+IPT,2,ICL)+1.
      IPAC2(LPC2+ICL)=1
      IF(NPA.LT.0)GO TO 380
C
C         PROCESS PARTICLE FROM THE SURFACE SOURCE.
      IF(JSU.GE.0)GO TO 60
      IF(NDET(IPT).EQ.0.AND.NDX(IPT).EQ.0)GO TO 375
      IPSC=12
      SWTM=WGT
      CALL STARTP
  375 JSU=ABS(JSU)
      IF(WC1(IPT).GT.0.)GO TO 60
      WCS1(IPT)=-WC1(IPT)*WGT
      WCS2(IPT)=-WC2(IPT)*WGT
      GO TO 60
C
C        SHORT LOOP IF PARTICLE IS UNCOLLIDED PART OF FORCED
C        COLLISION.
  380 D=MIN(DLS,DXL,DTC)
      DT=D/VEL
      PMF=HUGE
      JSU=NPA+1000000
      GO TO 190
C
C        THE HISTORY IS COMPLETE.
C        ADD THE TALLY DATA OF THIS HISTORY TO THE TOTAL
TALLY DATA.
  390 NCT(1)=NCT(1)+NCH(1)
      NCT(2)=NCT(2)+NCH(2)
      IF(MODE.EQ.2)GO TO 400
      SMUL(2)=SMUL(2)+SMUL(1)
      SMUL(3)=SMUL(3)+SMUL(1)**2
  400 IF(NTAL.GT.0)CALL TALSHF
      IF(NTAL.GT.0.AND.MOD(NPS,NPD).EQ.0)CALL ADDTFC
      RANK=RANI
      RANL=RANJ
      RETURN
C
*******************  PROCESS LOST PARTICLE.
*******************
*
        410 IF(KDB.GE.11)GO TO 450
C
C        CLEAR THE FIRST TALLY BLOCK AND READ VARIABLE COMMON
BACKUP.
      DO 420 I=1,MXF
  420 TAL(LTAL+I)=0.
      IF(NRSW.NE.0)CALL SUFWRT(0,ZERO)
      DO 425 I=1,NVARCM
  425 GVARCM(I)=GVBU(I)
      DO 430 I=1,LVARCM
  430 JVARCM(I)=JVBU(I)
C
C        RERUN HISTORY WITH FULL SURFACE SENSE CHECK AND
EVENT
C        PRINTING.
      IF(KRFLG.EQ.2.OR.NERR.GE.LOST(2))GO TO 440
      KRFLG=2
      NTII=0
      GO TO 40
C
C        RETURN TO PRINT DEBUG INFORMATION AND START A NEW
HISTORY.
  440 NERR=NERR+1
```

```
      IF(NERR.LE.LOST(2))KOV=1
      RETURN
C
******************  TERMINATE LONG HISTORY.
*******************
*
        450 NST=NST+256
      IF(NRSW.NE.O)CALL SUFWRT(O,ZERO)
      DO 460 I=1,NVARCM
  460 GVARCM(I)=GVBU(I)
      DO 470 I=1,LVARCM
  470 JVARCM(I)=JVBU(I)
      RANI=RANK
      RANJ=RANL
      IF(NSR.NE.6) then
          RETURN
      end if
      BACKSPACE IUSR
  480 BACKSPACE IUSR
      BACKSPACE IUSR
      NRRS=NRRS-1
      READ(IUSR)A
      IF(NPSR.EQ.ABS(A))GO TO 480
      NQSS=NQSS-1
      RETURN
      END


   SUBROUTINE TRANSM(DD,ST)
C         CALCULATE THE ATTENUATION AMFP OVER THE DISTANCE DD
FROM
C         XXX,YYY,ZZZ IN THE DIRECTION UUU,VVV,WWW.
C         ST IS THE RUSSIAN ROULETTE LEVEL.
      include 'CM.inc'
      include 'RC.inc'
C
      SD=0.
      AMFP=0.
      IF(LCA(LLCA+ICL).LT.0)CALL CHKCEL(ICL,3,J)
      FT=ST
      IF(FT.NE.0.)TT=-LOG(FT)
C
C         DO RUSSIAN ROULETTE ON SMALL SCORES.
   10 IF(FT.EQ.0.)GO TO 20
      IF(AMFP.LT.TT)GO TO 20
      T=EXP(-AMFP)
      IF(FT*RANG().GT.T)GO TO 30
      WGT=WGT*FT/T
      FT=T
      TT=AMFP
C
C         CALCULATE THE ATTENUATION FOR THIS SECTION OF THE
```

```
TRACK.
   20 CALL TRACK(ICL)
      IF(KDB.NE.0) then
         RETURN
      end if
      TOTM=0.
      IF(MLL(LMLL+1,ICL).EQ.0)GO TO 25
      IF(IPT.EQ.1)CALL ACETOT
      IF(IPT.EQ.2)CALL PHOTOT
   25 PLE=TOTM*RHO(LRHO+ICL)
C
      D=MIN(DLS,DD-SD)
C
*=====================================================================
*
*   THIS PORTION OF CODE CORRECTS FOR A VARIABLE DENSITY
*   ATMOSPHERE
*
*=====================================================================
*
C   CALCULATE A NEW MACROSCOPIC CROSS SECTION, PLE, BASED ON
C      A VARIABLE DENSITY ATMOSPHERE.
      NINP=1
C
C        TAKE RECIPROCAL OF MACROSCOPIC CROSS SECTION
     XMFP=1./PLE
      CALL EQDIST(XMFP, D, ZZZ, WWW, NINP)
      NINP=0
      PLE=1./XMFP
C
*
*=====================================================================
*
C        CALCULATE A NEW NUMBER OF MEAN FREE PATHS, AMFP.
      AMFP=AMFP+PLE*D
*
*=====================================================================
*
      IF(AMFP.GT.80.)GO TO 30
      SD=SD+DLS
C
C        UPDATE THE LOCATION AND THE NEW CELL.
      XXX=XXX+UUU*D
      YYY=YYY+VVV*D
      ZZZ=ZZZ+WWW*D
      DO 27 L=0,LEV-1
      UDT(1,L)=UDT(1,L)+UDT(4,L)*D
      UDT(2,L)=UDT(2,L)+UDT(5,L)*D
   27 UDT(3,L)=UDT(3,L)+UDT(6,L)*D
      TME=TME+D/VEL
      IF(SD.GE.DD) then
         RETURN
      end if
      JSU=JAP
      CALL NEWCEL(ZERO)
      IF(KDB.NE.0) then
         RETURN
      end if
      ICL=IAP
      IF(FIM(LFIM+IPT,IAP).NE.0.)GO TO 10
      CALL BEYOND(5)
C
C        RETURN WITH ZERO WEIGHT FOR ZERO IMP. OR IF SCORE IS
REJECTED.
   30 WGT=0.
      RETURN
      END
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
MARCH 1991

3. REPORT TYPE AND DATES COVERED
Technical

4. TITLE AND SUBTITLE
Modification of MCNP, Version 3B:
Incorporating a Variable Density Atmosphere

5. FUNDING NUMBERS

6. AUTHOR(S)
Monti, David Louis; Mathews, Kirk Alan

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology AFIT/ENP
Wright-Patterson AFB, OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/EN-TR-91-2

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Weapons Laboratory WL/NTN
Kirtland AFB, NM 87117

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

MCNP version 3B was modified to incorporate a continuously variable density atmosphere. This was accomplished by representing the variation of air density as a function of altitude with a series of continuous piecewise exponential curves up to a maximum altitude of 1000 km. User-written subroutines and functions were written which incorporated these piecewise functions. These subroutines and functions were subsequently incorporated into a production version of MCNP. Several MCNP subroutines and files were modified in support of these modifications. This report discusses detailed information regarding the theoretical development of the variable density model, the user-written subroutines and functions, the modifications to MCNP subroutines and files, and other relevant information.

14. SUBJECT TERMS
Variable Density Atmosphere, MCNP Modifications,
Variable Density Monte Carlo Transport

15. NUMBER OF PAGES
73

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED/UNLIMITED

20. LIMITATION OF ABSTRACT